Trabajo Fin de Máster

Identificación de lenguaje misógino a partir de minería de textos en redes sociales

Rubén Blanco Toledano

Tutoras:

Raquel Martínez Unanue

Lourdes Araujo Serna

Máster en Ingeniería y Ciencia de datos

UNED

Convocatoria: Junio

Curso 2020/2021

Agradecimientos

Gracias...

...a Raquel y Lourdes por su apoyo y su inestimable ayuda en el desarrollo del trabajo.

...a mi abuela Dolores, porque sin ella, sin sus coplas y sin su imaginación, la vida sería mucho más difícil.

...a mi abuela Maruja, que allá donde esté, la memoria le devuelva los bailes que se quedaron por el camino.

...a mi madre por darme un espacio de libertad, cariño y bondad en el que poder crecer y aprender.

Rubén Blanco Toledano

Madrid, 30 de junio de 2021

Índice general

Índice de figuras

Índice de tablas

Resumen

En este trabajo se estudia la creación de un identificador de agresividad y de mensajes de odio hacia mujeres (mensajes misóginos) a partir de datos recogidos de la red social *Twitter*. Se trata de una respuesta a la tarea planteada en *Automatic Misogyny Identification* (AMI) por IberVal 2018.

El estudio se compone de dos tareas. En la Task A se crea un identificador binario para determinar si un *tweet* tiene o no contenido misógino. En la Task B se desarrolla una clasificación acerca del entorno de los mensajes. Por un lado se identifica el objetivo al que va dirigido el mensaje (a una persona particular o a un público general) y por otro se realiza una clasificación del tipo de misoginia en torno a cinco posibles categorías de comportamientos misóginos.

Se ha utilizado un conjunto de datos en castellano y otro en inglés para realizar una comparación entre lenguas. El estudio se basa por una parte en el preprocesado de los datos y la vectorización de los textos (con métodos como *Bag of Words*, *TF-IDF* o *Doc2Vec*) y por otra en la búsqueda de los mejores clasificadores posibles (con modelos como *Naive Bayes*, *Regresión Logística*, *Support Vector Machine*, *Random Forest*, *Red Neuronal* y *Combinación de modelos*).

Palabras clave:

Minería de textos, Vectorización de textos, AMI, Automatic Misogyny Identification, Misoginia, Twitter, Clasifiación de Tweets, NLP

Abstract

This project studies the creation of an identification method of cases of aggressiveness and hate speech towards women (misogynistic messages) from Twitter data. This is a response to the Automatic Misogyny Identification (AMI) shared task at IberEval 2018

The study is composed of two tasks. Task A is a binary classification task to determine if a tweet has misogynous content or not. Task B is a classification task to study the message environment. On the one hand, the objective is to know if the misogyny message was purposely addressed to a specific target (a particular person) or not (general people). On the other hand, the purpose is to distinguish different types of misogyny according to five possible categories.

Two datasets are used in the study, one in Spanish and one in English, in order to make a comparison between languages. First of all, the study is based on the text vectorization (with *Bag of Words*, *TF-IDF* and *Doc2Vec* methods). Later, the objective is to search the best possible classifier algorithm (with models such as *Naive Bayes*, *Logistic Regression*, *Support Vector Machine*, *Random Forest*, *Neural Network* and *Combination of models*)

Key words:

Text mining, Text vectorization , AMI, Automatic Misogyny Identification, Misogyny, Twitter, Tweets classification, NLP

Capítulo 1

Introducción

1.1 Motivación

En la actualidad, en la era de las comunicaciones, las redes sociales tienen un papel muy importante puesto que facilitan que los usuarios publiquen y compartan contenido de forma que expresen sus opiniones y sentimientos. Dentro del campo del aprendizaje automático y el procesamiento de lenguaje natural, esta cantidad de datos ha permitido abordar tareas para comprender, medir y monitorizar algunos temas o eventos [Cambria et al., 2017].

Sin embargo, bajo este marco de libertad que ofrecen las redes sociales, los episodios de incitación al odio y el acoso han aumentado. Como consecuencia, se ha generado un interés en el campo de la inteligencia artificial y el procesamiento de lenguaje natural referentes a tareas relacionadas con cuestiones sociales y éticas. En este sentido, se está desarrollando una tendencia, "AI for social good", que tiene como objetivo el desarrollo de aplicaciones para maximizar los impactos sociales "buenos" y minimizar los impactos "malos". Un ejemplo de esta tendencia es el estudio de la detección de ideas suicidas con el fin de la intervención temprana [Gaur et al., 2019]. El objetivo de tendencias como "AI for social good" es generar respuestas positivas después de abordar el contenido abusivo alentando así a la comunidad a que adopte un enfoque proactivo para transformar entornos hostiles [Pamungkas et al., 2020].

Uno de estos casos de acoso en redes sociales es aquel que se da hacia las mujeres a través de mensajes misóginos. Según el informe *Pew Research Center Online Harass-*

ment [Duggan, 2017], en el entorno virtual las mujeres, y especialmente las mujeres jóvenes sufren formas de abuso sexual en tasas mucho mayores que los hombres. Alrededor del 21 % de las mujeres entre 18 y 21 años han sido acosadas sexualmente *online* frente al 9 % de hombres en el mismo rango de edad. Además, más de la mitad (un 53 %) de este grupo de edad de mujeres han recibido imágenes sexuales explícitas que no fueron solicitadas. El 35 % de las mujeres que han experimentado acoso sexual a través de redes sociales han catalogado estos hechos como extremos o muy molestos frente al 16 % de los hombres.

Por último, el 63 % de las mujeres piensan que las personas deberían sentirse bienvenidas y seguras en espacios *online* frente al 43 % de los hombres, mientras que los hombres (56 %) piensan que es más importante opinar libremente en las redes frente al 36 % de las mujeres.

En el estudio visto en [Braithwaite, 2014] se detalló que el seguimiento de los contenidos publicados en redes sociales es útil para la prevención de delitos sexuales. En concreto, se demostró que existía una correlación entre la tasa anual de violaciones y la cantidad de mensajes con lenguaje misógino en *Twitter*. Esto viene como consecuencia del hecho de que no hay un límite claro entre lo que sucede en el mundo *online* y en el mundo *offline* [Malpas, 2008].

Por todos estos motivos surge la necesidad de realizar un proyecto cuyo objeto sea el estudio de la difusión de mensajes misóginos en redes sociales y la forma de identificar dichos mensajes.

Uno de los primeros trabajos en los que se ha intentado detectar los mensajes misóginos de forma manual en *Twitter* es el trabajo de [Hewitt et al., 2016]. El objetivo del estudio no era encontrar ejemplos de *tweets* con mensajes abusivos, sino encontrar un contexto en el que el lenguaje misógino está presente y es usado.

Ante este problema social, en IberEval 2018 se propuso una tarea de Identificación Automática de Misoginia (AMI) que pretende discernir mensajes misóginos de no misóginos [Fersini et al., 2018].

1.2 ¿Qué es AMI?

AMI (*Automatic Misogyny Identification*) es una campaña iniciada por Anzovino, Fersini y Rosso en 2018 que propusieron un primer conjunto de datos de referencia que captaba hechos misóginos en *Twitter*. Este conjunto de datos es un punto de partida para la identificación automática de misoginia en redes [Pamungkas et al., 2020].

En la tarea AMI IberEval 2018 se propuso la identificación automática de misoginia en dos idiomas, castellano e inglés. A su vez, esta tarea general se dividía en dos subtareas de clasificación. La primera trata de clasificar cada uno de los *tweets* en si es un mensaje misógino o no. Una vez que se ha hecho esta clasificación se pasa a la segunda tarea que a su vez es doble. Una parte consiste en clasificar el tipo de comportamiento de aquellos mensajes con contenido misógino. Esta clasificación se puede agrupar en cinco tipos: estereotipo, dominación, desviación, acoso sexual y descrédito. La otra parte consiste en identificar el objetivo del mensaje como activo (dirigido a una persona concreta) o pasivo (dirigido a un grupo general) [Pamungkas et al., 2020].

Esta tarea tuvo respuesta por parte de once equipos diferentes de cinco países (España, Italia, Estados Unidos, Irlanda y Reino Unido) [Fersini et al., 2018]

Este es el marco de trabajo sobre el que se ha realizado todo el estudio. Se han tomado los conjuntos de datos anotados de tal forma que se pueda realizar un estudio basándose en técnicas de Aprendizaje Automático Supervisado.

1.3 Objetivos

El objetivo principal de este trabajo es realizar un estudio teórico-práctico de un problema de clasificación de mensajes misóginos a partir de datos que vienen de redes sociales, como *Twitter*.

A lo largo del desarrollo del trabajo se han intentado cumplir tres objetivos:

1. Estudiar un problema basado en datos de redes sociales. Estudiar posibles soluciones, herramientas y diferentes colecciones de datos de la bibliografía a partir de los cuales poder entender el contexto del problema en la actualidad.

2. Desarrollar diferentes sistemas de aprendizaje automático con los que poder identificar mensajes con sesgo misógino y ser capaz de caracterizarlo según una serie de características.

3. Evaluar de forma detallada la solución propuesta y comparar con colecciones de referencia.

1.4 Estructura de la memoria

En este apartado se va a hacer un pequeño resumen del contenido de las siguientes secciones del trabajo.

En el capítulo 2 se habla del proceso de modelado de los datos que se emplean en el estudio. Por un lado se habla del modelado de los datos en sí y por otro lado de los modelos y métodos empleados.

Lo primero que se hace es una descripción de los datos utilizados (sección 2.1) para posteriormente hacer un análisis general de los mismos (sección 2.2). En la sección 2.3 se explica todo el proceso de preprocesado de los datos, muy necesarios cuando se trabaja con datos que vienen de redes sociales. Finalmente se pasa a describir los tres modelos de vectorización de textos con los que se ha trabajado (sección 2.4) y los algoritmos de clasificación que se han aplicado para construir los diferentes clasificadores (sección 2.4).

A lo largo del capítulo 2 se han explicado los modelos desde un punto de vista teórico. En el capítulo 3 se muestran las herramientas concretas y los paquetes de *python* que han sido utilizados para desarrollar el estudio. En la sección 3.1 se habla de las técnicas correspondientes a la parte de preprocesado de los datos. En la sección 3.2 se comentan las relacionadas con la parte de vectorización de textos. Los paquetes empleados para aplicar los algoritmos de clasificación se muestran en la sección 3.3. Finalmente en la sección 3.4 se explican las dos métricas que se utilizan para estudiar los resultados (*accuracy* y estadístico F_1) para el caso de una clasificación binaria y una clasificación multiclase.

Una vez se ha desarrollado todo el proceso de estudio de las diferentes posibilidades de preprocesado y de clasificación, en el capítulo 4 se muestran todos los resultados que

se han obtenido y su correspondiente justificación. En la sección 4.1 se muestran los resultados correspondientes a la Task A (identificación de mensajes misóginos). En la sección 4.2 se muestran los resultados obtenidos para la Task B. Por una parte se habla de los resultados de la tarea de clasificación por objetivo (sección 4.2.1) y por otro lado de la tarea de clasificación según el tema (sección 4.2.2). Finalmente en la sección 4.3 se realiza una comparación con los resultados obtenidos por otros participantes de la competición AMI [Fersini et al., 2018].

Finalmente, en el capítulo 5 se detallan las conclusiones finales a las que se han llegado después de desarrollar todo el trabajo.

Capítulo 2

Modelado de los datos

En este apartado se van a detallar los aspectos técnicos del trabajo realizado. En el apartado 2.1 se va a realizar una descripción de los datos que se han usado: desde su origen, composición... En el siguiente apartado (2.2) se describe el estado general de los datos y su naturaleza. Posteriormente, en la sección 2.3 se explican los procesos de preprocesamiento que se han aplicado para realizar la tarea de limpieza de los datos. En el apartado 2.4 se explica los distintos métodos de vectorización de los textos para poder posteriormente hacer un análisis de aprendizaje automático a partir de los modelos que se ven en la sección 2.5

2.1 Descripción de los datos

Para poder llevar a cabo el proceso de crear tanto el identificador de mensajes misóginos como la clasificación de los mismos, se utilizan los datos que vienen dados de la tarea AMI [Fersini et al., 2018]. Se han tomado dos conjuntos de datos diferentes: un conjunto de *tweets* en castellano y otro en inglés. Estos datos han sido recogidos siguiendo tres enfoques diferentes:

1. Tomar los textos que incluyesen palabras claves como: "puta", "zorra", "bitch" o "whore"

2. Monitorizar cuentas que pueden ser potencialmente víctimas de ataques misóginos, por ejemplo, cuentas públicas de feministas o víctimas de *gamergate*

[VanDerWerff, 2014].

3. Descargar el historial de *tweets* de usuarios identificados como misóginos, por ejemplo, perfiles que de manera pública han declarado su odio hacia las mujeres.

Todos los textos usados pertenecen al espacio temporal comprendido entre el 20 de julio de 2017 y el 30 de noviembre de 2017, lo que da lugar a un *corpus* de 83 millones de mensajes en inglés y 72 millones en castellano. Posteriormente se tomó un subconjunto de estos textos [Fersini et al., 2018].

La etapa de etiquetado se ha realizado en dos partes: primero dos anotadores crearon y etiquetaron un "patrón de referencia". Los casos en los que había conflicto, un tercer colaborador experto en el tema resolvía el problema. En segundo lugar, los *tweets* fueron etiquetados a través de una votación realizada por colaboradores externos al estudio. El "patrón de referencia" realizado anteriormente se usaba como medida de control de calidad [Fersini et al., 2018].

Una vez la parte de etiquetado ha sido completada, los corpus en inglés y castellano se dividen en un conjunto de entrenamiento y otro de test. Para el conjunto de datos en castellano se tienen 3307 *tweets* para la fase de entrenamiento y 831 para la fase de test. Por el contrario, para el conjunto de datos en inglés se tienen 3251 *tweets* para la fase de entrenamiento y 726 para la fase de test [Fersini et al., 2018].

Todos estos mensajes presentes en el conjunto de entrenamiento vienen descritos por una serie de campos. Estos son:

- **id**: denota un identificador único del *tweet*
- **texto**: representa el contenido del mensaje
- **misogynous**: define si un *tweet* es misógino (y por tanto toma el valor 1) o si no lo es (y toma el valor 0)

Un ejemplo de un mensaje anotado como no misógino es:

> « *Feliz cumpleaños @USER, un año más siendo LEYENDA! Que tengas un día de puta madre y celébralo como te mereces! Un abrazo!* »

Y un ejemplo de mensaje anotado como misógino es:

« *YA CALLATE PUTA PERRA ME TIENES CANSADO CON TUS PENDEJADAS* »

- **misogyny category**: para un mensaje anotado como misógino, se define el tipo de comportamiento misógino. Este comportamiento puede ser categorizado en uno del siguiente conjunto de valores:

  - **stereotype**: define aquellos mensajes cuyo contenido hace referencia a un estereotipo o objetificación. Son mensajes que hablan de una imagen o una idea generalizada y simplificada de la mujer. Suelen estar relacionados con aspectos referentes al físico y tienden a realizar comparaciones con estándares impuestos.

  Un ejemplo de un *tweet* que potencia un estereotipo es el siguiente:

  « *Es la hora de la siesta. ¿ Cómo se dice 'siesta' en femenino? Fregar los platos.* »

  - **dominance**: denota aquellos *tweets* que ejercen dominación sobre la mujer. Son mensajes que afirman la superioridad de los hombres sobre la mujer con el fin de resaltar la desigualdad de género.

  Un ejemplo de este tipo de mensaje misógino es:

  « *@USER Que te calles so guarra.* »

  - **derailing**: definen mensajes que tienden a desviar la atención de un problema y con el fin de justificar el abuso, se rechaza la responsabilidad masculina. Es una forma de interrumpir la conversación y redirigirla a un ámbito más cómodo para los hombres.

  Un ejemplo de este tipo es el que se muestra a continuación

  « *Las personas feministas solo se dedican a expandir mierda sobre la cara de los hombres y despues realmente lo que buscan es igualdad* »

- **sexual harassment**: hace referencia a mensajes en los que está presente acciones de acoso sexual o amenazas violentas con el fin de afirmar el poder físico sobre las mujeres.

  « *@USER @USER Mmmmm la mas tetona y puta de Mexico, pinche puta tan buena . Mmmm te mereces k te usan, abusan en la cama, y te cogen como la puta k eres, mmmm mama una verga* »

- **discredit**: hace referencia a casos de desprestigio hacia una mujer. El fin de estos *tweets* es hablar mal de una mujer sin ningún otro objetivo.

  « *@USER tú eres una perra infernal* »

- **objetivo**: define a quién va dirigido el *tweet* misógino. Si el objetivo es una persona concreta o individual, toma el valor "active" y si denota varios potenciales receptores, es decir, si el mensaje es genérico, toma el valor "passive".

Un ejemplo de *tweet* anotado como "active" es el que se ve a continuación:

« *@USER Esta mujer es una perra.* »

En cambio, un ejemplo de un *tweet* anotado como "passive" es:

« *Las viejas de antes, puteaban bonito para ganar dinero. Las de hoy acusan de acoso o violacion a algún pobre y buen hombre.* »

Todos los ejemplos que se han visto pertenecen al conjunto de datos en castellano por simplicidad. Se podría hacer el estudio de forma totalmente análoga en el conjunto en inglés pudiendo encontrar ejemplos equivalentes. Además, con el fin de preservar la intimidad de las personas citadas en los *tweets*, se ha sustituido cada uno de los nombres por USER.

2.2 Estado general de los datos

En este apartado se va a hacer un pequeño recorrido por la naturaleza de los datos. Según las características vistas en la sección 2.1, se va a cuantificar cuántos de los *tweets* pertenecen a cada tipo para así caracterizar el conjunto de datos y conocer si los datos están balanceados o no.

Lo primero que se va a mostrar es una nube de palabras (Figura 1) en la que se muestran los términos más frecuentes que aparecen en el subconjunto de los *tweets* anotados como misóginos. Viendo sendas nubes de palabra es sencillo intuir cómo va a ser esa distribución de características. Palabras como “perra”, “puta”, “bitch” o “whore” están muy presentes, lo que hace pensar que hay una gran presencia de *tweets* de descrédito, puesto que en este tipo de mensaje los insultos suelen estar muy presentes.

Figura 1: Nube de palabras del conjunto de tweets misóginos. Arriba se muestra el resultado para el conjunto de datos en castellano y abajo para el conjunto de datos en inglés.

En la Tabla 1 se muestra el número de *tweets* dentro del conjunto de datos que pertenece a cada una de las características vistas con anterioridad. La representación tanto de mensajes misóginos como de mensajes no misóginos es parecida para ambos

casos, por lo que se puede afirmar que ambas clases están bien balanceadas en ambas lenguas. En el conjunto de datos de entrenamiento en castellano se tiene un 50.14 % de *tweets* no misóginos y un 49.86 % de *tweets* misóginos. Por otro lado, en el conjunto de datos de entrenamiento en inglés se tiene un 51.77 % de *tweets* no misóginos y un 48.23 % de *tweets* misóginos.

Tabla 1: Tabla en la que se resume el número de tweets que pertenecen a cada una de las características descritas en la sección 2.1. Se muestran los valores tanto para el conjunto de entrenamiento como para el conjunto de test. Cada uno de ellos está subdividido para cada lengua.

	Entrenamiento		Test	
	Castellano	Inglés	Castellano	Inglés
Misógino	1649	1568	415	283
No misógino	658	1683	416	443
Discredit	978	943	287	123
Sexual Harassment	198	410	51	32
Derailing	20	29	6	28
Stereotype	151	137	17	72
Dominance	302	49	54	28
Active	1455	942	370	104
Passive	194	626	45	179

En cambio, la distribución de las categorías que describen el tipo de mensaje misógino distan bastante de ser balanceadas. Tanto en el conjunto en castellano como en inglés, lo más común es encontrar mensajes con contenido de descrédito (tal y como se intuía en las nubes de palabra). Del conjunto total de *tweets* un 60 % pertenecen a la categoría descrédito para el conjunto de castellano y un 60.1 % para el conjunto en inglés.

Algo similar sucede cuando se centra la atención en el número de *tweets* que son enviados a una persona concreta (Active) respecto de los que son enviados en un contexto general (Passive). En este caso se evidencia más que las clases no están balanceadas. En el conjunto de datos en castellano un 11.8 % de *tweets* Passive frente al 88.2 % Acti-

ve. Y en el conjunto de datos en inglés se tiene que un 40 % de los mensajes son Passive frente al 60 % Active.

2.3 Preprocesado de los datos

En el contexto en el que se está trabajando, la parte de preprocesado de los *tweets* se basa en un procedimiento de limpieza y preparación de los textos que posteriormente van a ser clasificados. Se sabe que los textos no estructurados, como son los que vienen de Internet (y de *Twitter* en este caso), contienen una gran cantidad de ruido, es decir, datos que no contienen información útil para el análisis [Symeonidis et al., 2018]

Según [Fayyad et al., 2003], en un conjunto de datos, el porcentaje de ruido que puede estar presente en un texto se encuentra alrededor del 40 %, lo que puede llevar a grandes problemas a la hora de aplicar técnicas de aprendizaje automático. Además, en *Twitter* se tiende a hacer un uso de un lenguaje poco formal, por lo que es común encontrar errores tipográficos, abreviaciones, uso de jerga concreta, enfatización de las emociones con el uso abusivo de signos de puntuación, etc. Una idea para tratar esta situación es eliminar todos estos términos o sustituirlos por otros términos que los englobe.

Por todo esto, surge la necesidad de cribar y de realizar una tarea de preprocesamiento puesto que de la calidad de los datos depende el éxito del aprendizaje automático que se hace posteriormente.

Realizar el preprocesado de textos que vienen de *tweets* es una tarea bastante compleja y de vital importancia puesto que seleccionar la forma en la que se realiza el preprocesado tiene consecuencias directas en la posterior clasificación [Haddi et al., 2013]. En [Psomakelis et al., 2015], indican que la experiencia en la anotación automática de *tweets* basada en la detección de *emojis* es problemática porque no siempre refleja el sentimiento general expresado. Esto sucede especialmente cuando se trata de un mensaje neutral. Para el caso que se está estudiando en este trabajo, se va a mantener la presencia de estos *emojis* puesto que en el conjunto de datos que se tiene la presencia de mensajes de carácter neutral es pequeña en comparación con el conjunto de textos que tienen un objetivo muy claro.

En la Figura 2 se puede ver una pequeña muestra de los *emojis* que aparecen en los distintos *tweets*.

```
'🙃 ': 'upside-down face'}
'💕 ': 'two hearts'}
'😅 ': 'grinning face with sweat'}
'☝ ': 'index pointing up'}
'😍 ': 'smiling face with heart-eyes'}
'😂 ': 'face with tears of joy'}
'😍 ': 'smiling face with heart-eyes'}
'😏 ': 'smirking face'}
'💕 ': 'two hearts', '😌 ': 'relieved face'}
'👋 ': 'waving hand'}
'❤️ ': 'red heart'}
'🙆 ': 'person gesturing OK'}
'😡 ': 'pouting face'}
'😧 ': 'anguished face'}
'😰 ': 'anxious face with sweat'}
```

Figura 2: Subconjunto de emojis que se encuentran en el conjunto de datos.

A continuación se van a detallar los pasos que se han seguido para realizar esta tarea de preprocesamiento acompañada de algunos ejemplos. En estos ejemplos solo se va a tomar el caso de los datos en castellano por simplicidad. El procesamiento para los *tweets* en inglés es totalmente análogo al caso en castellano. Además, para mantener la privacidad de las personas implicadas, en el caso de que el mensaje contenga el nombre de usuario, se va a sustituir por @USER. Los pasos en el preprocesamiento son:

- Transformar todo el texto a minúsculas

En los *tweets* es normal encontrar palabras escritas totalmente en mayúsculas, palabras que intercalan mayúsculas y minúsculas... En consecuencia, para homogeneizar todos los mensajes, se transforman todos a minúsculas.

- Eliminar las *stop words*

Se eliminan las palabras más comunes en una lengua que no aportan significado por sí mismas [spaCy, 2021]. Pueden ser palabras como artículos, preposiciones, conjunciones... En inglés, por ejemplo, se tienen palabras como “and” o “I”. En el ejemplo que se muestra a continuación se ve como palabras como “la” o “bien” son eliminadas.

«Editar, **además de** complicado, **es lo que hace de algo que** merezca **la** pena o **no, porque** puedes **haber** grabado **la** puta hostia **de** planos, **que si no los** montas **bien**, **con un** sentido y **los** maquillas atractivamente, **se va** a **la** mierda.»	«Editar , complicado , merezca pena o , puedes grabado puta hostia planos , montas , sentido y maquillas atractivamente , a mierda .»

- Sustituir todos los *links* o *urls* por la palabra clave **URL**

Una forma de añadir más información al mensaje es añadiendo *links* que redireccionen a otras páginas. Esto a la hora de la clasificación no ayuda porque se trata como parte del mensaje. Por este motivo es necesario reemplazar estos enlaces por URL tal y como se ve en el siguiente ejemplo.

« Una violación cada 8 horas en españa Más de 1000 asesinatos machistas desde 2003 Todas hemos sufrido acoso callejero alguna vez Pero oye, son casos aislados y somos unas exageradas porque 'vivimos en uno de los mejores países del mundo' **https://t.co/mcSF1sGWQn** »	«violación 8 españa 1000 asesinatos machistas 2003 sufrido acoso callejero oye , casos aislados y exageradas ' vivimos mejores países mundo ' **URL** »

- Sustituir las menciones a otros usuarios por la palabra clave **MENTION**.

Otro elemento muy común en *Twitter* es la mención a otros usuarios de la red. Se ha tomado como mención aquellos términos que comienzan por "@", forma usada por *Twitter* para nombrar a alguien. A la hora de realizar la clasificación, el nombre concreto de la persona no es relevante. En el siguiente ejemplo se muestra uno de estos casos.

« **@USER** Tu eres una perra comunista! »	« **@USER** Tu eres una perra comunista! »

- Transformar loas signos de puntuación poco usuales:
  - **MULT_EXCLAMATION**: Concatenación de exclamaciones como "! ! ! ! ! ! !" o "!!!"
  - **MULT_INTERROGATION**:Concatenación de interrogaciones como "! ! ! ! ! ! !" o "????????????"
  - **MULT_PUNCTUATION**

En el caso de los signos de puntuación, lo primero que se hizo fue eliminar espacios innecesarios, es decir, se ha pasado de " !" a "!". Además, en el caso de que existiese una concatenación de signos de puntuación, se ha sustituido por una palabra clave según los casos vistos anteriormente. A continuación se muestra un ejemplo de uso de MULT_EXCLAMATION.

« @USER Regalé el libro a mi hija de 14 años y no veas qué éxito...Le está encantando. Gracias de corazón, Pamela. A seguir currando. Feliz año **!!!** »	« MENCIÓN Regalé libro a hija 14 años y veas éxito ... encantando . Gracias corazón , Pamela . A seguir currando . Feliz año **MULT_EXCLAMACIÓN** »

Un caso de MULT_INTERROGATION es:

« Por las mujeres tienen cuatro neuronas**???** Una para cada fogon »	« mujeres neuronas **MULT_INTERROGACION** fogon »

Y finalmente un ejemplo de la sustitución por MULT_PUNCTUATION es:

« @USER **¿¡¿**Un debate**?!?** No es sólo dar voz. Se ha de enseñar qué es un maltrato y por qué no se ha de cuestionar lo que es un maltrato. Las señales en la piel se ven, las que dejan en la mente no.»	« MENCIÓN **MULT_PUNCTUACIÓN** debate **MULT_PUNCTUACIÓN** voz . enseñar maltrato y cuestionar maltrato . señales piel ven , dejan mente . »

- Sustituir los *hashtags*

  - **MISO_HASHTAG**: Hace referencia a la aparición de *hashtags* con contenido misógino o usados con intención misógina. Algunos ejemplos son: #todasputas #cállateyfriega o #slut
  - **FEM_HASHTAG**: Hace referencia a la aparición de *hashtags* con contenido feminista o usados con intención feminista de apoyo. Algunos ejemplos son: #metoo, #niunamas o #endviolence
  - **HASHTAG**: Se usa para el resto de *hashtags* que no se agrupan en ninguno de los dos conjuntos anteriores.

En el caso del uso de *hashtags*, estos elementos se clasifican según las tres categorías mostradas anteriormente cada una con su palabra clave correspondiente. Un ejemplo de sustitución en el caso de un *hashtag* misógino se ve a continuación.

« El biribiri de las estupidas adolescentes manipuladas por la **#femimoda** aburre a la sociedad y enoja a los abusadores. **#feminazis**»	« biribiri estupidas adolescentes manipuladas **MISO_HASHTAG** aburre a sociedad y enoja a abusadores . **MISO_HASHTAG** »

En el caso de que un *hashtag* feminista esté presente, la sustitución es similar a la que se ve en el siguiente ejemplo.

« El riesgo constante de acoso y violación para las niñas y adolescentes migrantes y refugiadas se produce en toda su trayectoria por parte de agresores que incluyen proveedores de servicios y personas que deberían protegerlas. **#endviolence #NiUnaMenos** https://t.co/MkTJJwxG5H https://t.co/DOgYsdFjSf»	« riesgo constante acoso violación niñas adolescentes migrantes refugiadas produce trayectoria agresores incluyen proveedores servicios personas deberían protegerlas . **FEM_HASHTAG FEM_HASHTAG** URL URL »

Finalmente, el resto de *hashtags* que no se pueden agrupar en ninguna de las categorías anteriores se sustituye por la palabra HASHTAG de forma análoga a la que se muestra en el siguiente ejemplo.

« Bien joder una puta alegría!! Te lo mereces pequeña **#OTGala9** »	« joder puta alegría MULT_EXCLAMACIÓN mereces pequeña **HASHTAG** »

Una vez realizada toda esta tarea de limpieza y preprocesado de los datos, se pasa a entrenar algunos modelos de aprendizaje automático. Para la tarea de entrenamiento no es posible usar directamente el texto de los *tweets*, sino que es necesario que se realice una transformación para que sean entradas válidas en los modelos de entrenamiento. A esto se le conoce como vectorización de textos.

2.4 Proceso de vectorización

Las instancias de entrenamiento que se introducen en los modelos de aprendizaje automático no pueden ser textos. Por tanto, es necesario realizar una tarea de transformación de los mismos. Para ello se hará uso de unos procesos de vectorización.

En este trabajo se ha trabajado con tres de ellos: *Bag of words* (sección 2.4.1), *TF-IDF* (sección 2.4.2) y *Doc2Vec* (sección 2.4.3).

2.4.1 Bag of words

El modelo *bag of words* o bolsa de palabras es una forma de representar textos cuando se utilizan en algoritmos de aprendizaje automático. Es una forma de extraer características de los textos para el posterior entrenamiento. El tipo de representación que se obtiene es simple y flexible. Es una representación de un texto con la que se describe la ocurrencia de una palabra dentro de un documento. Esto está relacionado con dos aspectos: un vocabulario de palabras conocido y la medida de la presencia de cada término o palabra. El nombre de *bag of words* viene dado porque toda informa-

ción sobre la estructura de las palabras dentro del documento se ve descartada. Solo se tiene en cuenta cuántas veces aparece un término en el documento y no dónde. Intuitivamente se dice que dos documentos serán similares si su contenido también lo es. La complejidad de este método viene de la propia definición del diccionario de palabras. [Brownlee, 2017].

Cada documento se transforma en un vector de términos que aparecen en el mismo. La identificación de los términos en el documento puede basarse en las palabras presentes en el texto, que es lo que se conoce como *bag of words* o bolsa de palabras. [Martins et al., 2003].

En la representación de términos por el método *bag of words*, un término puede ser representado por palabras simples (1-gramas) o por varias palabras (en general, n-gramas). El conjunto de documentos puede ser representado en forma de tabla o matriz según la forma que se ve en la Tabla 2

Tabla 2: Representación matricial de los términos de un documento según el modelo bag of words

	t_1	t_2	$\cdots$	t_m
d_1	$a_{1,1}$	$a_{1,2}$	$\cdots$	$a_{1,m}$
d_2	$a_{2,1}$	$a_{2,2}$	$\cdots$	$a_{2,m}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
d_n	$a_{n,1}$	$a_{n,2}$	$\cdots$	$a_{n,m}$

En la Tabla 2 se representan n documentos y cada uno de ellos está compuesto por m términos. Por tanto, cada documento puede ser entendido como un vector $d_i = (a_{i,1}, a_{i,2}, \cdots, a_{i,m})$ y cada valor $a_{i,j}$ hace referencia al peso del término t_j en el documento d_i. Si se toma este peso como un valor binario, se puede decir que $a_{i,j} = 1$ si t_j está presente en d_i y $a_{i,j} = 0$ en caso contrario. La forma en la que se define el peso puede variar. Por ejemplo, se puede considerar el número de veces que aparece el término en el documento [Martins et al., 2003].

2.4.2 TF-IDF

TF-IDF (*Term frequency-inverse focument frequency*) es un método que se usa para dar un peso a cada palabra dentro del texto de acuerdo a la singularidad o a la presencia de la palabra en el texto, es decir, recoge la relevancia de las palabras en un texto [Yun-tao et al., 2005]

Es un método que se utiliza como factor de ponderación en la recuperación de información en minería de textos. Por convención, el valor de TF-IDF aumenta proporcionalmente al número de veces que una palabra aparece en un documento, pero se compensa según la frecuencia de la palabra en el corpus. Esto ayuda a controlar el hecho de que algunas palabras son más comunes que otras. El término relativo a la frecuencia inversa de los documentos es una medida de si el término es común o raro en todos los documentos [Christian et al., 2016].

La idea básica de este algoritmo es representar cada documento, D, como un vector en un espacio vectorial (expresión 1). De esta forma, documentos que tengan un contenido similar estarán descritos por vectores similares. Cada dimensión del espacio vectorial representa un término (una palabra o n-grama) [Joachims, 1996].

$$D = (t_i, t_j, ..., t_k) \tag{1}$$

donde cada t_k representa un término contenido en un documento D [Salton and Buckley, 1988].

El método TF-IDF viene de la composición de dos términos: **TF** (*Term frequency*) e **IDF** (*inverse document frequency*).

TERM FRECUENCY (TF)

Se puede decir que un término que tenga una gran frecuencia en un documento tiende a describir mejor al texto que otro menos frecuente. En consecuencia este término recibirá una puntuación de importancia mayor. Un mecanismo de puntuación será contabilizar cuántas veces aparece un término en un documento. De esta forma se asigna a cada término del documento un peso que depende del número de apariciones. La for-

ma más simple es definir el peso como el número de veces que el término t aparece en el documento d. Este esquema de peso es el que se conoce como *term frequency* (o término de frecuencia), $tf_{t,d}$ [Schütze et al., 2008].

Es conocido que el término de frecuencia mejora la exhaustividad, pero no siempre mejora la precisión. Esto se debe a que los términos más frecuentes tienden a aparecer en muchos textos lo que aporta poco poder de discriminación. Para evitar esto, los términos con frecuencia muy alta suelen eliminarse del conjunto de términos. Encontrar el umbral óptimo es uno de los problemas que surge con este método [Tokunaga and Makoto, 1994].

El conjunto de pesos TF, con cualquier función de ponderación que haga referencia al número de ocurrencias de cada término, puede ser interpretado como un compendio cualitativo de las palabras más relevantes del documento. Con todo esto, parece intuitivo que dos documentos con representaciones similares de palabras tienen un contenido similar [Schütze et al., 2008].

$$TF = f_k/N \tag{2}$$

***INVERSE DOCUMENT FRECUENCY* (IDF)**

Contabilizar la importancia de los términos como se hace en el apartado anterior tiene un problema: trata a todos los términos con igual importancia al tratar de evaluar la relevancia. Para evitar este hecho, se introduce un la idea de un mecanismo que consiga atenuar el efecto de términos recurrentes. Una idea es reducir el peso de términos con alta frecuencia definida como el número total de apariciones en el documento. Se trata de reducir el peso tf por un factor que crece con la frecuencia de recogida [Schütze et al., 2008].

Para este objetivo es usual usar la frecuencia de documento dft, que mide el número de documentos de la colección que contienen el término t. De forma intuitiva, si un término aparece en pocos documentos que contienen el término será más importante que el caso que el término aparezca en muchos documentos, pues ese término será mucho más general [Schütze et al., 2008].

Por tanto, se define el *inverse document frequency* de un término t como se ve en la expresión 3-

$$IDF_t = \log\left(\frac{N}{df_t}\right) \tag{3}$$

donde N hace referencia al número total de documentos y df_t es la frecuencia del término t por documento.

De esta forma, un término poco usual tiene un valor de IDF_t alto mientras que para términos muy frecuentes el valor será bajo [Schütze et al., 2008]. Como el término IDF representa lo específico que es un término, es esperado que mejor la precisión [Tokunaga and Makoto, 1994].

***TERM FREQUENCY - INVERSE DOCUMENT FRECUENCY* (TF-IDF)**

Si ahora se combinan las dos métricas anteriores, se da lugar a la definición de un peso compuesto para cada término en el documento: $TF-IDF$, cuya expresión se puede ver en la ecuación 4 [Schütze et al., 2008]. Esta idea viene descrita por primera ven en [Salton, 1989].

$$TF-IDF_{t,d} = TF_{t,d} \cdot IDF_t \tag{4}$$

El término $TF-IDF$ asigna a cada término un peso tal que [Schütze et al., 2008]:

1. Es mayor cuando el término t aparece muchas veces en una pequeña cantidad de documentos dando así un alto poder de discriminación de estos documentos.

2. Es más pequeño cuando el término aparece pocas veces en un documento o aparece en muchos documentos. Esto indica cuando el término tiene una relevancia menos pronunciada.

3. Es mínimo cuando el término aparece en prácticamente todos los documentos.

Los factores TF e IDF contribuyen a la mejora tanto de la exhaustividad como de la precisión respectivamente [Tokunaga and Makoto, 1994].

2.4.3 Doc2Vec

En [Mikolov et al., 2013] se propone un método de representación de palabras basado en redes neuronales llamado *Word2Vec* (Figura 3). Este método, dada una secuencia de palabras de entrenamiento $\{w\}_i = \{w_1, w_2, ..., w_T\}$, busca tener una representación vectorial para cada palabra. Para conseguir esto, el objetivo es maximizar la función de verosimilitud que se muestra en la expresión 5 [Kim et al., 2019].

$$\frac{1}{T}\sum_{t=k}^{T-k} logp(w_t|w_{t-k},\cdots,w_{t+k}) \tag{5}$$

donde k es el tamaño de la ventana para conservar la información del contexto. Es bastante usual que las predicciones correspondan a clasificación multiclase, por lo que se suele hacer uso de la función softmax (ecuación 6).

$$p(w_t|w_{t-k},\cdots,w_{t+k}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}} \tag{6}$$

donde cada y_i es el valor de la salida i de una red neuronal calculado tal y como se ve en la expresión 7

$$y = b + Uh(w_{t-k},\cdots,w_{t+k},W) \tag{7}$$

donde b es la desviación entre la capa oculta y la capa de salida de la red neuronal, U es la matriz de pesos entre la capa oculta y la capa de salida, h es la media o concatenación de palabras del contexto y W es la matriz de palabras.

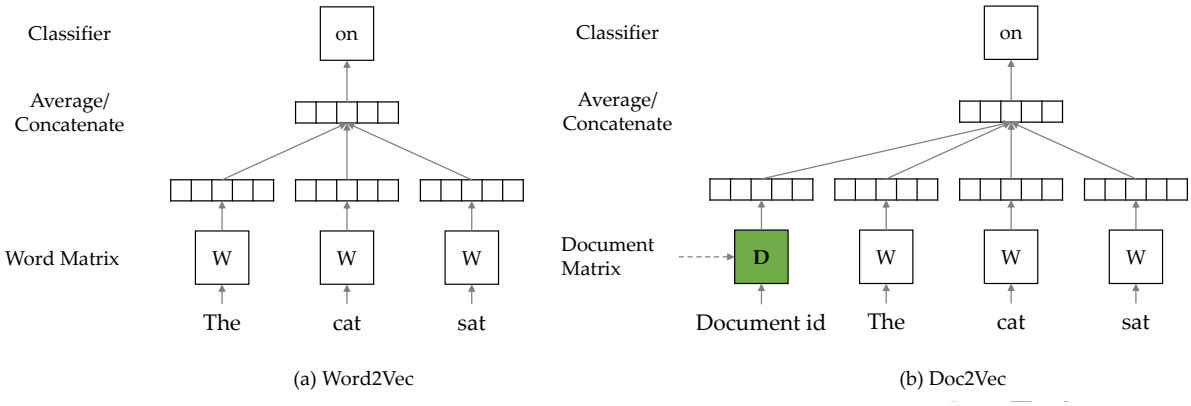

Figura 3: Representación gráfica de los modelos Word2Vec (figura de la izquierda) y Doc2Vec (figura de la derecha). Extraído de [Kim et al., 2019].

En la sección 2.5.5 se comenta en profundidad la idea de la red neuronal, pero en la Figura 4 se muestra una representación gráfica de dicha red neuronal para el modelo *Word2Vec*. En la capa de entrada se encuentra el vector de palabras con su correspondiente matriz de palabras W. En la capa o capas ocultas está presente la matriz de pesos respecto de la capa de salida y en la capa de salida se tiene el resultado de cada y_i.

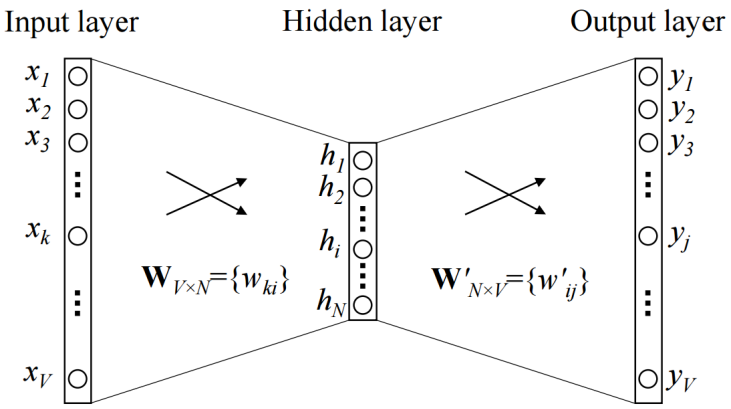

Figura 4: Representación de una red neuronal correspondiente a un modelo Word2Vec. Extraído de [Rong, 2014]

Doc2Vec es una extensión del modelo *Word2Vec* que tiene como objetivo determinar un vector adecuado para un párrafo con el fin de mantener las relaciones semánticas entre varios documentos. Al igual que *Word2Vec* cada palabra está representada por un vector continuo d-dimensional tal que $d << V$ (con V el tamaño del vocabulario en el corpus). Además, el documento también está representado por un vector en el mismo espacio vectorial como se ve en la Figura 3 [Kim et al., 2019].

En el método *Doc2Vec*, cada documento se ve referenciado por un único vector que está representado por una columna de la matriz D mientras que cada término está referenciada a un único vector representado por una columna en la matriz W. Por tanto, la modificación que se realiza en la formulación de la red neuronal es añadir la matriz D tal y como se ve en la expresión 8 [Kim et al., 2019].

$$y = b + Uh(w_{t-k}, \cdots, w_{t+k}, W, D) \tag{8}$$

Una vez la red está bien entrenada, se puede obtener una representación de cada documento en el corpus.

Por todo esto, el método de vectorización *Doc2Vec* es considerado como un método muy usado para predecir una palabra dadas las otras palabras dentro de un contexto. Una de las aplicaciones más comunes es usarlo como forma de predicción de la siguiente palabra en una oración [Le and Mikolov, 2014].

2.5 Modelos de clasificación

En esta apartado se van a detallar los modelos de aprendizaje automático que se han usado para conseguir tanto el identificador de mensajes misóginos como la clasificación de mensaje según el tema y el objetivo.

2.5.1 Naive Bayes

El método de clasificación Naive Bayes está basado en los principios de la estadística bayesiana. Se parte de un conjunto de características $\mathbf{X} = \{X_1, ..., X_n\}$ que son usadas para el proceso de aprendizaje. Se puede generar un grafo dirigido en el que cada nodo se ve representado por una característica y cada arista representa la relación entre dos variables o características. Cuando dos nodos están conectados, al formar un grafo dirigido, se puede diferenciar entre un nodo padre (origen) y un nodo hijo (destino) [Taheri and Mammadov, 2013].

A continuación se denota X_i como cada una de las características y $P_a(X_i)$ el conjunto de variables a los que está unido el nodo X_i. Dada esta estructura, se puede conocer la distribución de probabilidad asociada a cada variable [Taheri and Mammadov, 2013], que viene dada por la expresión 9.

$$P(X) = \prod_{i=1}^{n} P(X_i | P_a(X_i)) \tag{9}$$

En el método de clasificación Naive Bayes, las características X_i se toman independientes dada una clase, es decir, cada característica tiene únicamente una clase como padre [Taheri and Mammadov, 2013]. Como consecuencia de esto se asume que cada característica solo depende de la clase como se puede ejemplificar en la Figura 5.

Si ahora se denota como C a la clase y $\mathbf{X}$ el conjunto de características observadas,

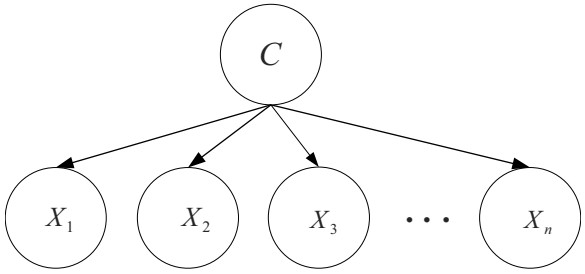

Figura 5: Representación gráfica de un clasificador Naive Bayes. Extraído de [Taheri and Mammadov, 2013].

predecir la clase a la que pertenece $\mathbf{X}$ pasa por usar la regla de Bayes vista anteriormente (expresión 10).

$$P(C|\mathbf{X}) = \frac{P(C)}{P(\mathbf{X})}P(\mathbf{X}|C) \tag{10}$$

Como las características $X = \{X_1, X_2, .., X_n\}$ son condicionalmente independientes unas de las otras, la probabilidad condicional $P(\mathbf{X}|C)$ puede describirse a través de las X_i. Teniendo esto en cuenta, se puede llegar fácilmente a la expresión 11.

$$P(C|\mathbf{X}) = \frac{P(C)}{P(\mathbf{X})}\prod_{i=1}^{n} P(X_i|C) \tag{11}$$

En problemas de clasificación, como es el caso, la ecuación 11 es suficiente para predecir cuál es la clase más probable a la que pertenece una instancia que viene caracterizada por un conjunto de características observadas. Para estimar la probabilidad de la clase $P(C)$ y las probabilidades condicionales $P(X_i|C)\ \forall i \in [1,n]$ existen distintos modelos y aproximaciones que intentan dar una solución [Taheri and Mammadov, 2013].

Este método es robusto cuando se observan grandes desviaciones de sucesos reales o esperados en el conjunto de datos. Además, es un modelo fácil de implementar y útil en el caso de conjuntos de datos con alta dimensionalidad [Taheri and Mammadov, 2013].

2.5.2 Regresión Logística

El modelo de regresión logística es un enfoque de modelado que se utiliza para describir la relación de varias características X_i a una variable dependiente, por lo general, dicotómica [Kleinbaum et al., 2002].

La función que define este modelo es la conocida función logística tal y como se muestra en la expresión 12).

$$f(z) = \frac{1}{1+e^{-z}} \tag{12}$$

El rango de la función es $R = (0,1)$. Como consecuencia, el significado que se toma es que el modelo logístico devuelve la probabilidad de pertenencia a una clase. Otra de las características de este modelo es la forma de su función, como se ve en la Figura 6. Se trata de una forma de sigma. Existe un valor umbral z_{min} antes del cual pequeñas variaciones en z no conllevan grandes variaciones. Posteriormente se ve que pequeños cambios en z dan lugar a una variación muy rápida del valor de la función hasta que se llega a otro valor límite, z_{max}, a partir del cual los cambios vuelven a ser poco perceptibles [Kleinbaum et al., 2002].

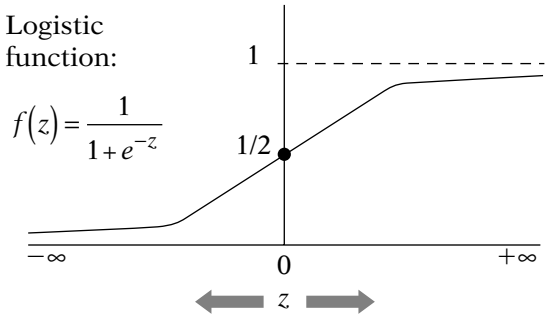

Figura 6: Representación gráfica de la función logística. Extraído de [Kleinbaum et al., 2002].

Una vez se ha visto el comportamiento de la función logística, se puede pasar a describir el propio modelo. Se define z como una combinación lineal de las características X_i tal y como se ve en la expresión 13.

$$\begin{aligned} z &= \alpha + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n \\ &= \alpha + \sum_{i=1}^{n} \beta_i X_i \end{aligned} \tag{13}$$

donde los términos α y β_i constantes y las características X_i son independientes.

Si ahora se toma la expresión 12 y en ella se sustituye la expresión 13, se consigue la que se conoce como la expresión del modelo logístico (ecuación 14).

$$f(z) = \frac{1}{1 + e^{-(\alpha + \sum_{i=1}^{n} \beta_i X_i)}} \tag{14}$$

Los valores de los parámetros α y β_i son desconocidos a priori. A pesar de ello, pueden ser calculados con los datos que provienen del conjunto de entrenamiento.

Finalmente, se concluye que la probabilidad de que dado un conjunto de características $\mathbf{X} = \{X_1, ..., X_n\}$ que definen un elemento pertenezcan a la clase C puede ser descrita por la expresión 15.

$$P(C|X_1, X_2, ..., X_n) = \frac{1}{1 + e^{-(\alpha + \sum_{i=1}^{n} \beta_i X_i)}} \tag{15}$$

2.5.3 SVM

Los modelos basados en *Support Vector Machine*, SVM, son un conjunto de métodos de aprendizaje supervisado que utilizan como hipótesis el uso de funciones lineales en un espacio vectorial de características de dimensión alta [Jakkula, 2006].

El punto de partida de los métodos SVM es que existe alguna dependencia desconocida (y por lo general no lineal) entre algún vector de entrada X de alta dimensión y una salida escalar (o un vector de salida en el caso de SVM multiclase) tal que $y = f(X)$. Se desconoce la información acerca de las funciones de probabilidad conjunta subyacentes del sistema. Por esto, la única forma disponible que se tiene es un conjunto de datos de entrenamiento $D = \{X_i, y_i\} \in X \text{ x } Y$, $\forall i = [1, 2..., l]$ donde l representa el número de datos de entrenamiento y por tanto es igual al tamaño del conjunto de entrenamiento D [Wang, 2005].

Los SVM son llamados modelos "paramétricos", no porque no tengan parámetros, sino porque los parámetros no están predefinidos y su número viene dado por los datos de entrenamiento. De esta forma, la capacidad del modelo viene impulsada por la complejidad de los datos [Wang, 2005].

Los modelos SVM están implementados siguiendo la idea de que se mapea una entrada de vectores en un espacio vectorial Z de características de alta dimensión a través de algún mapeo no lineal elegido a priori. En este espacio se construye una superficie

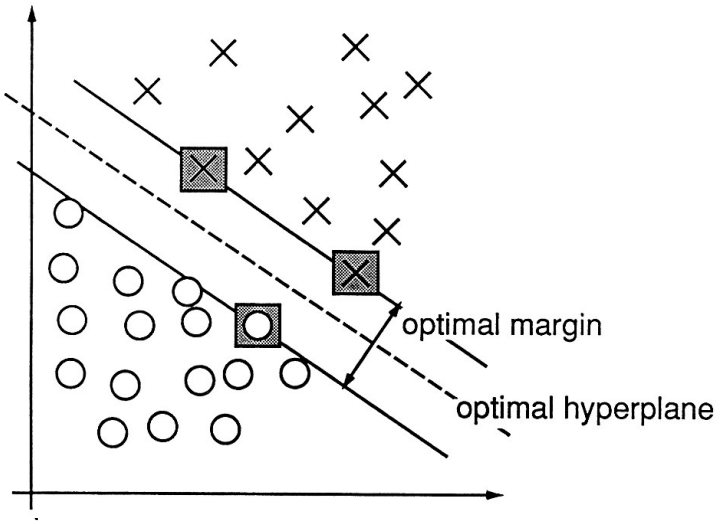

Figura 7: Representación gráfica de SVM en la que se ejemplifica el hiperplano óptimo. Extraído de [Cortes and Vapnik, 1995].

de decisión lineal con propiedades que garantizan una alta capacidad de generalización de la red. Ante este planteamiento surgen dos problemas. El primero es cómo encontrar un hiperplano que generalice correctamente la dimensionalidad del espacio de entrada. El otro problema está relacionado con la computación. Computacionalmente, tratar espacios de tan alta dimensión es muy costoso [Cortes and Vapnik, 1995].

El primer problema se solucionó con la idea de hiperplano óptimo para clases separables recogida en [Vapnik, 2006]. Se define un hiperplano óptimo como la función de decisión lineal con margen máximo entre los vectores de las dos clases, tal y como se ve en la Figura 7. Se vio que para construir estos hiperplanos óptimos solo es necesario tener en cuenta una pequeña cantidad de los datos de entrenamiento, llamados vectores de soporte, que determinaban el margen. Además, se demostró que si los vectores de entrenamiento están separados sin errores por un hiperplano óptimo, el valor esperado de la probabilidad de cometer un error viene descrito por la expresión 16.

$$E[P(\text{error})] \leq \frac{E[\text{número de vectores de soporte}]}{\text{número de vectores de entrenamiento}} \tag{16}$$

Como se ve en [Vapnik, 1963], existe en el campo de la teoría del aprendizaje estadístico un teorema que afirma la existencia de un hiperplano óptimo. Si se define la distancia desde el hiperplano de separación al vector de soporte más cercano como el margen del hiperplano, la SVM selecciona el hiperplano de separación de margen má-

ximo. Esta elección maximiza la capacidad para predecir la clasificación correcta. El teorema asume que los datos sobre los que se entrena la SVM se extraen de la misma distribución de datos sobre la que se prueba. Además, una SVM no asume que los valores de los datos de entrenamiento siguen una distribución normal [Noble, 2006].

Hasta ahora se ha asumido que los datos pueden ser separados a través de una línea recta, pero hay datos que no se pueden separar de forma tan clara. Para estos casos, el algoritmo puede ser modificado para añadir lo que se conoce como un margen suave. Esto permite que algunos puntos se encuentren en la clase equivocada y esto no afecte al resultado final. No es deseable que la SVM admite demasiados errores de clasificación. Por esto, se introduce un parámetro por el usuario que especifica cuántos ejemplares erróneos se permiten [Noble, 2006].

2.5.4 Random Forest

El modelo *Random Forest* es un método basado en un conjunto de árboles de decisión en el que cada árbol es dependiente de una colección de variables aleatorias. Formalmente, se tiene un vector de características n-dimensional $\mathbf{X} = \{X_1, X_2, ..., X_n\}$ que representan los valores reales de las entradas o variables predictoras. Además se dispone de una variable aleatoria Y que representa el valor real de la respuesta [Cutler et al., 2012].

El objetivo, al igual que en los modelos anteriores, es encontrar la función de predicción $f(X)$ a partir de la cual se permite obtener Y dado el conjunto de características $\mathbf{X}$. Esta función viene determinada por la función de pérdida $L(Y, f(X))$ la cual está definida de forma que el valor esperado del error (expresión 17) minimice ecuación [Cutler et al., 2012]:

$$E_{XY}\left(\, L(Y, f(X))\,\right) \tag{17}$$

De forma intuitiva, la función de pérdidas, $L(Y, f(X))$, es una medida de cuánto se parecen $f(X)$ e Y. Es usual que para el caso de estudio de regresión se tome como función de pérdida el error cuadrático (expresión 18) y para el caso de clasificación se use una función delta de Dirac (expresión 19) [Cutler et al., 2012]:

$$L(Y, f(X)) = (Y - f(X))^2 \tag{18}$$

$$L(Y, f(X)) = I(Y \neq f(X)) = \begin{cases} 0 & \text{si } Y = f(X) \\ 1 & \text{otro caso} \end{cases} \tag{19}$$

Como en el caso que se está estudiando, el objetivo final es obtener un clasificador (tanto para la Tarea A como para la Tarea B), a partir de aquí se va a centrar la atención únicamente en el caso de clasificación.

Al minimizar la función de pérdidas $E_{XY}(L(Y, f(X)))$ se obtiene la expresión 20 donde $\mathscr{Y}$ hace referencia al conjunto de posibles valores de salida Y. Es sencillo comprobar que el resultado al que se llega es la regla de Bayes que se ha visto anteriormente (expresión 9) [Cutler et al., 2012].

$$f(X) = \arg\max_{y \in \mathscr{Y}} P(Y = y | X = x) \tag{20}$$

Los conjuntos construyen la función $f(\mathbf{X})$ a base de los conocidos aprendices de base $\{h_1(x), h_2(x), ..., h_k(x)\}$, que se combinan para dar lugar a $f(x)$ de la forma que se muestra en la expresión 21.

$$f(X) = \arg\max_{y \in \mathscr{Y}} \sum_{j=1}^{J} I(y = h_j(x)) \tag{21}$$

En los modelos Random Forest, el aprendiz de base j-ésimo es un árbol que se denota $h_j(X, \Theta_j)$ donde Θ_j es un conjunto de variables aleatorias independientes entre sí [Cutler et al., 2012].

Los árboles usados en Random Forest se basan en una iteración recursiva con una salida binaria. El predictor sigue una secuencia de particiones binarias según variables individuales. En este proceso iterativo, se denomina nodo raíz a aquel del que parte todo el predictor. Los nodos que no se dividen se llaman nodos terminales y forman la última partición del predictor. Cada nodo no terminal se divide en dos descendientes

[Cutler et al., 2012].

Si las variables son categóricas, X_i toma valores de un conjunto finito de categorías $S = \{s_{i,1}, s_{i,2}, ..., s_{i,m}\}$. La forma en la que se realiza la división entre las categorías se elige considerando las posibles divisiones de cada variable de predicción y eligiendo el mejor según algún criterio [Cutler et al., 2012].

En el contexto de clasificación, donde hay K clases, un criterio típico de división es el índice de gini (expresión 22).

$$Q = \sum_{k \neq k'}^{K} \hat{p}_k \hat{p}_{k'} \tag{22}$$

donde $\hat{p}_k$ es la proporción de clase k observada en el nodo tal y como se ve en la ecuación 23:

$$\hat{p}_k = \frac{1}{n} \sum_{i=1}^{n} I(y_i = k) \tag{23}$$

El criterio de división se puede tratar como una medida de la bondad de la clasificación. Si se denotan como Q_L y Q_R a los candidatos que quedan en el nodo de la izquierda y de la derecha respectivamente y a n_L y n_R sus respectivos tamaños, se toma la división que minimiza la expresión 24:

$$Q_{split} = n_L Q_L + n_R Q_R \tag{24}$$

2.5.5 Red Neuronal

Las redes neuronales artificiales se basan en el modelo de red neuronal del cerebro humano en el que existen nodos (neuronas) conectados entre sí que transmiten información entre ellos mismos [Shanmugamani, 2018].

La idea más simple de un modelo de red neuronal artificial es la idea de neurona artificial o perceptrón (Figura 8). Se trata de una neurona que toma varias entradas y con ellas realiza una suma ponderada dando lugar a una salida binaria (expresión 26). El peso de cada entrada se determina durante el proceso de entrenamiento. Una vez realizada la suma, el resultado pasa por una función (en el caso de la figura, una

función escalón que da lugar a una clasificación binaria) que determina el valor de la salida [Shanmugamani, 2018].

$$y_i = \begin{cases} 1 & \text{si } \sum_{i=0}^{m} w_i x_i + \varepsilon_i = X_i^T \mathbf{w} + \varepsilon > 0 \\ 0 & \text{otro caso} \end{cases} \tag{25}$$

donde $X \in \mathbb{R}^p$ hace referencia al conjunto de características, $\mathbf{w}$ hace referencia a los pesos de cada instancia, m el número de entradas y $\varepsilon \sim N(0, \sigma^2)$ una medida de la desviación (independiente de los valores de entrada) [Ramasubramanian and Singh, 2017].

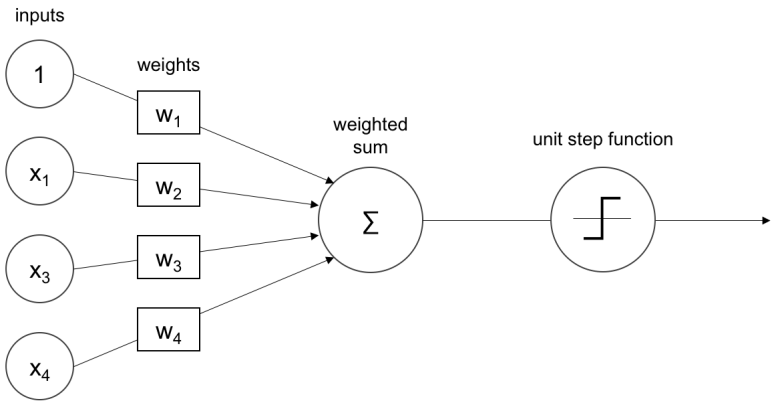

Figura 8: Representación gráfica de un perceptrón. Extraído de [Shanmugamani, 2018].

Para una salida binaria, como este caso, el valor de y_i puede ser modelado a partir de una regresión logística y como se ha visto anteriormente, se asume que la probabilidad de pertenencia a una clase viene descrita por la expresión 26 [Pang et al., 2020].

$$p(y_i = 1|X_i) = \frac{1}{1 + e^{-X_i^T \beta}} \tag{26}$$

Para entrenar el modelo es necesario definir una función de pérdida con el fin de penalizar los errores cometidos por el modelo. Para una regresión lineal, una función de pérdida muy usada es el error cuadrático (expresión 27). Sin embargo para una regresión logística se suele usar la función logarítmica de verosimilitud [Pang et al., 2020]. Posteriormente en la sección 2.5.5.1 se detallarán diferentes tipos de estas funciones de pérdidas.

$$L(y_i, X_i; \beta_) = (y_i - X_i\beta)^2 \tag{27}$$

El objetivo final será minimizar la media de la función de pérdida sobre todas las instancias $\frac{1}{n}\sum_{i=1}^{n} L(y_i, X_i; \beta)$. Para casos sencillos, como la regresión lineal, este problema tiene una solución analítica. Pero para otros más complejos necesita de un algoritmo iterativo [Pang et al., 2020].

En general, los modelos de redes neuronales están diseñados para resolver tareas de regresión y clasificación, pero en lugar de utilizar una función lineal, se puede usar otras funciones algo más sofisticadas con representaciones más flexibles [Pang et al., 2020]. Algunas de estas funciones (funciones de activación) se detallarán en la sección 2.5.5.2

La idea del perceptrón simple solo es útil para clasificaciones lineales. Por tanto, se necesita expandir la arquitectura de red neuronal para incluir más perceptrones que hagan algo más que una regresión lineal. Esta expansión se puede ver en la Figura 9. A este tipo de red se le suele llamar perceptrón multicapa (MLP) . Se compone de una capa de entrada (capa de la izquierda), una capa de salida (capa de la derecha) y un conjunto de capas ocultas que se corresponden con el resto de capas que están entre la de entrada y salida [Ramasubramanian and Singh, 2017].

Cada capa se puede representar por un vector que se obtiene multiplicando la capa anterior por una matriz de pesos más un vector de desviación, de forma análoga al perceptrón simple. Finalmente cada resultado se transforma acorde a una función no lineal (algunos ejemplos de estas funciones se verán en la sección) [Pang et al., 2020]. Cada capa puede tener una función de activación que se elige dependiendo del objetivo del problema [Shanmugamani, 2018].

Según el diseño, el número de neuronas de la capa de entrada y salida está definido, pero encontrar el diseño de las capas ocultas no es tan sencillo [Ramasubramanian and Singh, 2017].

En el proceso de entrenamiento se determinan estos valores de peso y sesgo. Los valores del modelo se inicializan con valores aleatorios al comienzo del entrenamiento. El error se calcula usando una función de pérdida. Según el error, los pesos se ajustan en cada paso. El entrenamiento se detiene cuando el error no se puede reducir más o

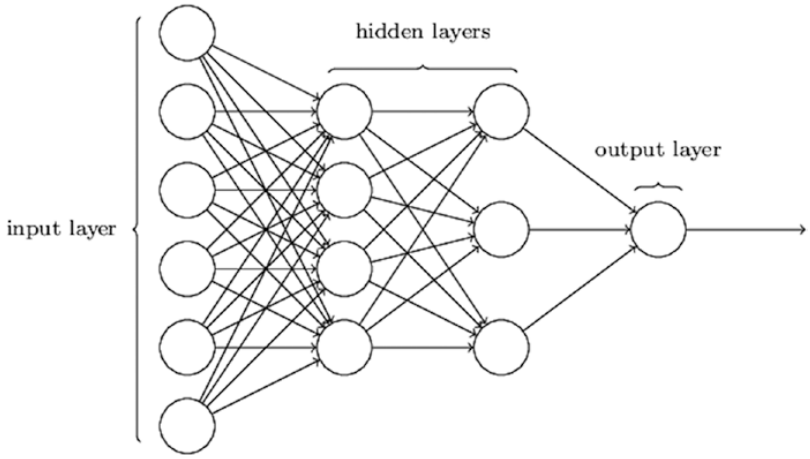

Figura 9: Representación gráfica de un modelo de red neuronal. De izquierda a derecha se muestra la capa de entrada, el conjunto de capas ocultas y la capa de salida. Extraído de [Ramasubramanian and Singh, 2017].

cuando llega a un límite dado [Shanmugamani, 2018].

Con toda esta información, las redes neuronales artificiales se pueden resumir en tres componentes principales que configuran el ejercicio de entrenamiento:

1. Arquitectura: elementos como el número de capas, matriz de pesos, desviación, conexiones...

2. Funciones de activación: Se refiere al mecanismo de cómo se comportan las neuronas en consecuencia de las señales de las demás.

3. Tasa de aprendizaje: forma en la que los pesos de la red neuronal cambian con el tiempo.

2.5.5.1 Arquitectura

Dentro del apartado arquitectura, se engloban muchas características como el número de neuronas que hay en cada capa, el número de capas, la matriz de pesos... Entre estas características está la función de pérdida, que mide el error que se está cometiendo en el proceso de entrenamiento. Hay varias posibilidades a partir de la cual definir la función de pérdida:

- **Softmax**. Obliga a la red neuronal a que la suma de pesos esté normalizada. Por tanto, los valores de salida de la función softmax se pueden considerar como parte

de una distribución de probabilidad. Es útil en problemas de clasificación multiclase. Convierte las salidas en probabilidades dividiendo la salida por la suma de todos los demás valores. La distancia euclidiana se puede calcular entre las probabilidades del softmax para la optimización [Shanmugamani, 2018].

- **Entropía cruzada**. Compara la distancia entre las salidas de softmax con la codificación binaria. Es una función de pérdida para la que se debe minimizar el error. Las redes neuronales estiman la probabilidad de los datos dados para cada clase. La probabilidad debe utilizar para la estabilidad numérica. Maximizar una función equivale a minimizar el negativo de la misma función. [Shanmugamani, 2018].

- **Abandono**. Es una forma de homogeneizar las redes neuronales para evitar el sobreajuste. Durante el entrenamiento, la capa de abandono paraliza el proceso al eliminar las unidades ocultas de forma estocástica. También es una forma de combinar varias redes neuronales. [Shanmugamani, 2018].

- **Normalización por lotes**. Aumenta la estabilidad y el rendimiento del entrenamiento de redes neuronales. Normaliza la salida de una capa con media cero y desviación. Esto reduce el sobreajuste y hace que el entrenamiento sea más rápido. Es útil en el caso de redes complejas [Shanmugamani, 2018].

- **Regularización L1 y L2**. L1 penaliza el valor absoluto del peso y tiene a hacer que el valor de todos los pesos sea cero. L2 penaliza el valor del cuadrado del peso y tiende a reducir el peso durante el entrenamiento. Ambas regularizaciones asumen que los modelos con pesos pequeños son mejores [Shanmugamani, 2018].

2.5.5.2 Funciones de activación

El papel de las funciones de activación es hacer que una red neuronal no sea solamente lineal. Una vez en la última capa se tiene el resultado de pasar por todas las capas ocultas, a través de la función de activación se puede conocer el resultado de la clasificación o de la regresión. La existencia de un mayor número de posibilidades hace que se pueda aprender redes con dinámicas más complejas [Shanmugamani, 2018].

Algunos ejemplos de función de activación son:

- **Sigmoide**. Se trata de una función escalón suavizada, y por tanto diferenciable. Es usada para convertir cualquier valor en probabilidad y así usarse en clasificación binaria [Shanmugamani, 2018]. Gráfica de la izquierda de la Figura 10.
- **Tangente hiperbólica**. Es la versión escalada de la función sigmoide. Asigna la entrada a un valor en el rango $[-1, 1]$. Los gradientes son más estables que en el caso anterior. También dispara de forma continua lo que hace que la red sea muy pesada [Shanmugamani, 2018]. Gráfica central de la Figura 10.
- **ReLU**. La función de activación ReLU (Rectified Linear Unit) tiene como entrada una gran cantidad de números. Esto hace que algunas neuronas no se activen al quedarse obsoletas. Esto aumenta la escasez, que es algo bueno. A las entradas negativas le asigna el valor 0 y las entradas positivas toman su valor. Esta función no dispara todo el tiempo y por tanto se puede entrenar más rápido. Es la opción menos costosa computacionalmente debido a la simpleza de la función [Shanmugamani, 2018]. Gráfica de la derecha de la Figura 10.

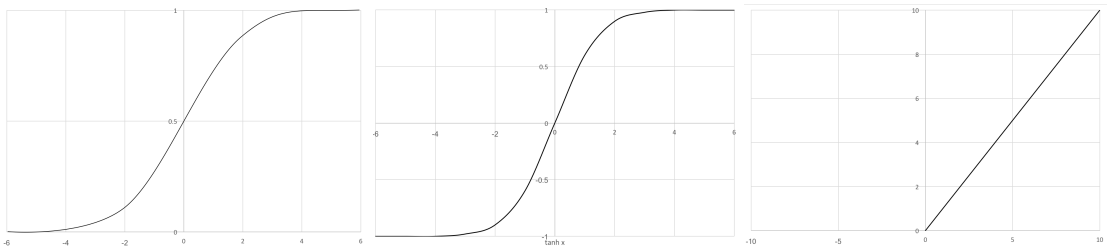

Figura 10: Representación gráfica de tres funciones de activación. De izquierda da derecha: Sigmoide, Tangente hiperbólica y ReLU . Extraído de [Ramasubramanian and Singh, 2017].

La elección de la función de activación depende en gran medida de la aplicación para la que se va a usar. Sin embargo, ReLU funciona bien en una amplia gama de problemas [Shanmugamani, 2018].

2.5.5.3 Tasa de aprendizaje

El proceso de entrenamiento de una Red Neuronal Artificial es complejo porque tiene varios parámetros para optimizar. El proceso de actualización de los pesos se denomina retropropagación. Y el procedimiento para minimizar el error se llama optimización.

Un algoritmo de retropropagación se usa normalmente para entrenar redes neuronales artificiales. Es un proceso de actualización de los pesos. Los pesos se actualizan desde el final hasta el principio en función del error calculado. Una vez calculado el error, se puede usar el descenso del gradiente para calcular la actualización de dicho peso 11.

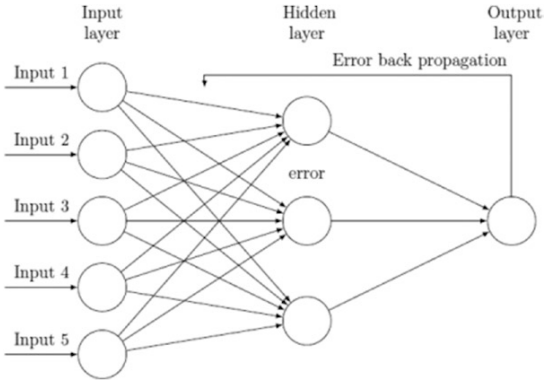

Figura 11: Representación gráfica del procedimiento de retropropagación. Extraído de [Ramasubramanian and Singh, 2017].

El descenso de gradiente es un algoritmo que realiza una optimización multidimensional. Es una técnica de optimización que se utiliza para mejorar la predicción del modelo. Una de las implementaciones es el descenso de gradiente estocástico (SGD). Implica calcular el valor del error y cambiar el valor de los pesos de forma que el error sea mínimo. La dirección para encontrar ese mínimo es la dirección en la que el gradiente (la derivada) de la función de pérdida sea más negativo.

La tasa de aprendizaje determina cómo de grande debe de ser cada paso de la iteración. SGD funciona mejor para optimizar funciones de pérdida no convexas.

SGD es un método similar al método de descenso de gradiente, pero en este caso solo usa datos parciales para entrenar cada iteración.

2.5.6 Combinación de modelos

Hasta ahora todos los modelos que se han visto tomaban como entrada un conjunto de características para posteriormente dar como salida un resultado que mostrase a qué clase pertenece. En cambio, los métodos de combinación de modelos son algo diferente.

Los métodos de combinación de modelo toman como entrada varios modelos predictores con el fin de obtener una mejor predicción que la que se obtendría con el mejor predictor individual [Géron, 2019]. Un esquema en el que se indica cuál es el funcionamiento de este tipo de algoritmos se puede ver en la Figura 12. Hay diferentes métodos de combinación de modelos, pero en este caso se va a centrar la atención en el modelo de votación.

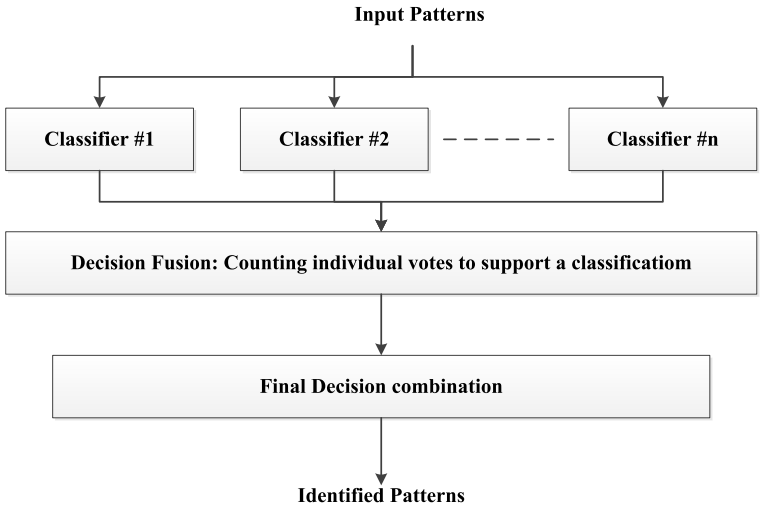

Figura 12: Representación gráfica en la que se explica el funcionamiento de los algoritmos de combinación de modelos. Extraído de [Mohandes et al., 2018].

A la hora de tomar una decisión acerca del algoritmo que es usado para calcular la salida Y, se pueden tomar dos opciones: hacer uso de un algoritmo con umbral estricto (*hard voting*) o un algoritmo de umbral flexible (*soft voting*).

Los algoritmos que usan *hard voting* toman como solución la salida del clasificador que ha pasado inmediatamente después por un umbral previamente establecido. En cambio, en los algoritmos de combinación de modelos con *soft voting*, se usa la estimación de probabilidad a posteriori para cada clasificador de entrada y posteriormente se toma la opción más probable [Mohandes et al., 2018].

Hay tres versiones de voto: unánime, más de la mitad de los votos y la mayoría de votos. Si se considera el vector de salida del clasificador i, $[d_{i,1}, d_{i,2}, ..., d_{i,N}]^T \in [0,1]^N$ donde $i \in [1, M]$ y $d_{i,j} = 1$ si las características del clasificador D_i se corresponde con una de las clases y 0 en otro caso [Mohandes et al., 2018].

El algoritmo de mayoría de voto en una decisión para una clase se ve en la expresión 28.

$$\sum_{i=1}^{M} d_{i,k} = \max_{j=1}^{N} \sum_{i=1}^{M} d_{i,j} \tag{28}$$

donde M es el número total de clasificadores, que suele ser un número impar, N es el número total de clases. Este tipo de modelos da una precisión cuando $M/2+1$ clasificadores dan una buena clasificación [Mohandes et al., 2018].

Capítulo 3

Metodología

En esta sección se van a enumerar las herramientas que se han usado para poder llevar a cabo todos los algoritmos y procesamientos que se han nombrado en la sección 2.

En la sección 3.1 se muestran las herramientas de *Python* que han sido usadas para desarrollar todo el procedimiento de limpieza y preprocesado de los *tweets*. En la sección 3.2 se muestran las herramientas usadas para el proceso de vectorización de textos.

Además, en la sección 3.4 se han descrito las métricas que se han usado para la caracterización de cada modelo: la métrica *accuracy* y el estadístico F_1.

3.1 Preprocesado

El paquete *spaCy* se ha usado a la hora de realizar el procesado de los textos. *spaCy* es una librería *open-source* usada para el estudio de NLP con *Python*. Está diseñado específicamente para crear aplicaciones que sean capaces de procesar y entender grandes cantidades de textos. Puede ser usado como sistema de extracción de información o comprensión del lenguaje natural o para realizar el preprocesamiento de textos [Honnibal and Montani, 2017].

Con el fin de realizar la nube de palabras que se ha visto en la Figura 1, se hace uso de la herramienta *wordcloud*. A través de un texto de entrada, se genera una imagen en la que las palabras más frecuentes aparecen en un tamaño mayor y las menos comunes

en un tamaño menor [Mueller, 2020].

Pare obtener una buena visualización de los *emojis* presentes en los *tweets* se ha hecho uso del paquete *demoji* de *Python*, que permite encontrar y eliminar estos elementos de un conjunto de textos [Solomon, 2020].

Para la eliminación de los términos conocidos como *stopwords* se ha usado la herramienta *NLTK, Natural Language Toolkit*. Se trata de un conjunto de módulos de *Python* que permite realizar tareas de procesamiento de textos. Además incluye algunos ejemplos, corpus, lectores, árboles, estructuras... También proporciona implementaciones de referencia para tokenizadores, lematizadores, etiquetadores, analizadores sintácticos, agrupadores, clasificadores, etc [Loper and Bird, 2002].

3.2 Vectorización

Una vez se ha realizado el proceso de pre-preprocesado de los datos, se pasa a indicar las herramientas utilizadas para la vectorización de los textos.

En el caso del método de vectorización TF-IDF se ha hecho uso de la herramienta *TfidfVectorizer* presente en el paquete *sklearn.feature_extraction.text* de *Python*. Esta herramienta transforma el conjunto de *tweets* en una matriz con representación TF-IDF [Buitinck et al., 2013] siguiendo lo explicado anteriormente, en el capítulo 2.4.2

Para el caso del método de vectorización *bag of words* se ha hecho uso de la herramienta CountVectorizer que proviene del paquete *sklearn.feature_extraction.text* de *Python* [Buitinck et al., 2013].

Finalmente, para en el caso de la vectorización usando *doc2vec*, se han utilizado las funciones *Doc2Vec* y *TaggedDocument* que vienen del paquete *gensim.models.doc2vec* [Řehůřek, 2020].

De esta forma, los datos de entrada de cada uno de los algoritmos de clasificación, es decir, los datos empleados para el entrenamiento de los modelos, vienen definidos directamente por el conjunto de *tokens* extraídos del proceso de vectorización de los *tweets*. Anteriormente, los *tweets* han pasado por un proceso de limpieza y preprocesamiento según lo que se ha visto en la sección 2.3 con las herramientas descritas en la sección 3.1.

3.3 Algoritmos de clasificación

Una vez se ha realizado el proceso de vectorización de los textos, se pasa a realizar la clasificación de los mismos. Para realizar esta tarea, como se ha visto en el Capítulo 2.5, se va a hacer uso de diversos algoritmos. La mayoría de ellos pueden encontrarse dentro del módulo *sklearn* de *python*, que integra una amplia gama de algoritmos de aprendizaje automático para problemas supervisados y no supervisados [Buitinck et al., 2013].

De esta forma:

- Para el algoritmo de Naive Bayes se ha hecho uso de la función *GaussianNB* del paquete *sklearn.naive_bayes*. El hiperparámetro *var_smoothing* se ha usado para la exploración del mejor clasificador. Este valor está relacionado con la varianza añadida al resto con el fin de llegar al cálculo de estabilidad [scikit-learn developers, 2020b].

- Para la regresión logística se ha hecho uso de la función *LogisticRegression* del paquete *sklearn.linear_model*. Se han usado hiperparámetros como *C* (que indica un parámetro de regularización) y *solver* (que indica el algoritmo de optimizción empleado) para la exploración del mejor clasificador [scikit-learn developers, 2020c].

- Para el algoritmo de *Support Vector Machine* se ha hecho uso de la función *SVC* del paquete *sklearn.svm*. En este caso se ha realizado una búsqueda en torno a hiperparámetros como *C* (que indica un parámetro de regularización), *kernel* (que indica el algoritmo de kernel usado) o *gamma* (que indica el coeficiente del algoritmo de kernel) [scikit-learn developers, 2020e].

- Para el algoritmo de *Random Forest* se ha hecho uso de la función *RandomForestClassifier* que viene del paquete *sklearn.ensemble*. Algunos de los hiperparámetros empleados para la búsqueda del mejor clasificados son *max_depth* (que indica la profundidad máxima a la que se permite llegar en el árbol),

max_features (que indica el número máximo de características que pueden describir cada división) o *n_estimators* (que indica el número de árboles en el bosque) [scikit-learn developers, 2020d].

- Para el algoritmo de *Red Neuronal* se ha hecho uso de la función *Keras* que viene del paquete *tensorflow* [Bisong, 2019]. Se ha hecho uso de una red neuronal secuencial con las capas de tipo *Dense*. La capa de entrada se compone de ocho neuronas mientras que la salida se compone de una para los estudios de la Task A y la clasificación según objetivo de la Task B y de cinco para la clasificación según tema de la Task B.

  A la hora de estudiar las posibilidades de los hiperparámetros, se ha buscado el número de capas ocultas, el número de neuronas de cada copa oculta y la tasa de aprendizaje tales que se obtuviese el mejor modelo. Los valores concretos para cada uno de los casos estudiados se detallan en el Anexo.

  Tanto para la capa de entrada como para el conjunto de capas ocultas, se ha tomado como función de activación *ReLU*. En el caso de las clasificaciones binarias (como son las correspondientes a las Task A y a la clasificación según objetivo de la Task B), se ha empleado como función de activación para la capa de salida la función *Sigmoide*. En el caso de la clasificación multiclase, como es el caso de la clasificación según tema de la Task B, se ha empleado como función de activación para la capa de salida la función *Softmax*.

  La función de pérdida que se ha empleado en todos los casos para la construcción de la red neuronal es la función *entropía cruzada*.

- Para el algoritmo de Combinación de modelos se ha hecho uso de la función *VotingClassifier* de la función *sklearn.ensemble* [scikit-learn developers, 2020a]. Se ha utilizado el tipo de votación *soft voting* junto con una versión de voto basado en la mayoría de voto

En el Anexo se muestran todos los detalles de las concretas con las que se han obtenido cada uno de los resultados de cada uno de los algoritmos aplicados.

3.4 Medida de estadísticos: *Accuracy* y Estadístico F1

Una vez se ha obtenido el resultado de la clasificación para cada instancia con los diferentes modelos, se puede pasar a comparar estos resultados con los resultados anotados del conjunto de test.

En dos de los tres estudios que se van a realizar la salida es binaria. En el caso de la Task A en misógino o no misógino y en el caso del estudio de objetivo en la Task B en activo o pasivo. En el otro caso (clasificación según tema en la Task B) la clasificación se realiza alrededor de cinco categorías, es decir, una clasificación multiclase.

3.4.1 Clasificación binaria

Para este tipo de casos binarios, el resultado de la comparación de cada resultado individual puede ser recogido en una de estas cuatro opciones [Chicco and Jurman, 2020]:

1. **Verdaderos Positivos o *True Positives* (TP)**: Un elemento positivo ha sido clasificado como positivo de forma correcta.

2. **Falso Negativo o *False Negative* (FN)**: Un elemento positivo ha sido clasificado como negativo de forma errónea.

3. **Verdadero Negativo o *True Negative* (FN)**: Un elemento negativo ha sido clasificado como negativo de forma correcta.

4. **Falso Positivo o *False Positive* (FN)**: Un elemento negativo ha sido clasificado como positivo de forma errónea.

Si esto se representa de forma matricial se obtiene la conocida matriz de confusión (expresión 29):

$$M = \begin{pmatrix} TP & FN \\ FP & TN \end{pmatrix} \tag{29}$$

De forma intuitiva se puede ver que el resultado de una buena clasificación binaria será aquel que tenga grandes valores para TP y TN menores para FP y FN [Chicco and Jurman, 2020].

Por simplicidad se introducen algunas métricas que ayudan a la comprensión de estos valores y que permitan describir la calidad de la predicción. Para realizar esta comparación se van a usar dos métricas: *accuracy* y el estadístico F1 tal y como se hace en la tarea AMI ([Fersini et al., 2018]).

ACCURACY

La métrica *accuracy* representa la relación que existe entre el número de resultados clasificados de forma correcta y el número total de resultados encontrados. Por tanto, está métrica se puede calcular como se ve en la expresión 30.

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{30}$$

Esta métrica tiene un rango $[0,1]$ teniendo 0 cuando no se ha clasificado bien ningún resultado y 1 cuando todos los resultados han sido correctamente clasificados, es decir, una clasificación perfecta.

La métrica *accuracy* falla cuando se intenta estimar la clasificación en conjuntos de datos no balanceados. Se define el *ni*, (*no-information error* como se ve en la expresión 31 que es una medida de cómo de balanceado está el conjunto de datos [Chicco and Jurman, 2020]:

$$ni = \frac{\text{máx}\{n^+, n^-\}}{n^+ + n^-} \tag{31}$$

Para un conjunto perfectamente balanceado $ni = 1/2$ y $ni >> 1/2$ para conjuntos no balanceados.

Esta sobre-representación de una clase sobre otra hace que la métrica precisión pierda algo de sentido [Chicco and Jurman, 2020].

ESTADÍSTICO F_1

El estadístico F_1 es una métrica muy usada que define cómo la media armónica entre la precisión y la exhaustividad tal y como se ve en la expresión 32

$$F_1 = \frac{2TP}{2TP+FP+FN} = 2\frac{precision \cdot recall}{precision + recall} \tag{32}$$

Al igual que el caso anterior, toma un valor 0 cuando la clasificación es errónea para todos los resultados y 1 cuando es una clasificación perfecta.

Una de las diferencias con la métrica *accuracy* es que no depende del número TN y por tanto no es simétrico para el intercambio de clases. En caso de que $TP = 0$ se toma el valor del estadístico como $F_1 = 1$.

3.4.2 Clasificación multiclase

Una vez se han conocido las métricas de evaluación para una clasificación binaria, se pasa a generalizar estas medidas para el caso en el que se tenga más de dos categorías. Esto se debe a que la clasificación multiclase necesita de métricas diferentes a la de la clasificación binaria [Tsoumakas and Katakis, 2007].

En general se tiene un conjunto de resultados con $L > 1$ características (si $L = 2$ se vuelve a la clasificación binaria) de forma que para $L > 2$ el problema se conoce como un problema de clasificación multiclase.

Se denota D al conjunto de datos completo de la evaluación, de forma que $(x_i, Y_i) \forall i = 1, 2, \cdots, |D|$. Las clases Y_i pueden ser definidas como $Y \subseteq L$ [Tsoumakas and Katakis, 2007].

Ahora, si denotamos a H como un clasificador multiclase, se definen los elementos resultantes de la predicción como $Z_i = H(x_i)$ [Tsoumakas and Katakis, 2007].

Una vez se han hecho la generalización de los elementos anteriores se puede pasar a definir las métricas.

ACCURACY

La métrica *accuracy* para una clasificación multiclase viene dada por la expresión 33 [Tsoumakas and Katakis, 2007].

$$accuracy(H,D) = \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \tag{33}$$

ESTADÍSTICO F1

En el caso del estadístico F_1 existen tres formas diferentes de definir esta métrica.

$\boxed{F_{1macro}}$

Se calcula tomando la media armónica de $precision_{macro}$ (expresión 34) y $recall_{macro}$ (expresión 35) [Vani and Rao, 2019].

$$precision_{macro} = \frac{1}{|L|}\sum_{i}^{|L|}P_i = \frac{1}{|L|}\sum_{i}^{|L|}\frac{TP_i}{TP_i + FP_i} \tag{34}$$

$$recall_{macro} = \frac{1}{|L|}\sum_{i}^{|L|}R_i = \frac{1}{|L|}\sum_{i}^{|L|}\frac{TP_i}{TP_i + FN_i} \tag{35}$$

Uniendo todo, la expresión para la métrica F_{1macro} se muestra en la expresión 39.

$$F_{1macro} = 2\frac{precision_{macro} \cdot recall_{macro}}{precision_{macro} + recall_{macro}} \tag{36}$$

$\boxed{F_{1micro}}$

Se calcula tomando la media armónica de $precision_{micro}$ (expresión 37) y $recall_{micro}$ (expresión 38) [Vani and Rao, 2019].

$$precision_{micro} = \frac{\sum_{i}^{|L|} TP_i}{\sum_{i}^{|L|} TP_i + \sum_{i}^{|L|} FP_i} \tag{37}$$

$$recall_{micro} = \frac{\sum_i^{|L|} TP_i}{\sum_i^{|L|} TP_i + \sum_i^{|L|} FN_i} \tag{38}$$

Uniendo todo, la expresión para la métrica F_{1micro} se muestra en la expresión **??**.

$$F_{1micro} = 2\frac{precision_{micro} \cdot recall_{micro}}{precision_{micro} + recall_{micro}} \tag{39}$$

$F_{1average}$

La métrica $F_{1average}$ se calcula tal y como se ve en la expresión 40

$$F_{1average} = \frac{1}{\sum_{i=1}^{|L|} N_i} \sum_{i=1}^{|L|} F_i N_i \tag{40}$$

donde $F_1, F_2, \cdots, F_{|L|}$ son los valores de los F_1 calculados para cada clase y $N_1, N_2, \cdots, N_n$ son el número de instancias de cada clase [Vani and Rao, 2019].

Capítulo 4

Resultados

Una vez detallado el proceso de experimentación y las diversos métodos de clasificación, se pasa a mostrar los resultados obtenidos siguiendo lo visto en el capítulo 3. Como se ha visto antes, el objetivo es doble: ser capaz de identificar si un mensaje es misógino o no (sección 4.1) y una vez hecha la clasificación, ser capaz de identificar en los mensajes misóginos su tipo dentro de la clasificación (sección 4.2).

Finalmente en la sección 4.3 se ha realizado una comparación con los resultados que ha obtenido un conjunto de participantes de la competición AMI para así poder tener una idea de la calidad de la clasificación realizada.

Para calcular los valores de las métricas resultantes se ha hecho uso de la herramienta *cross validation*. El error que acompaña a los valores de las métricas se corresponde con el error que aparece en el conjunto de iteraciones de esta validación cruzada.

4.1 Identificación de mensajes misóginos (*Task A*)

En la Tabla 3 se muestran los valores de la métrica *accuracy* y del estadístico F_1 para todos los modelos entrenados tanto para el conjunto de datos en castellano como para el conjunto de datos en inglés.

En la Figura 13 se muestran los gráficos de barras correspondientes al valor de la métrica *accuracy* (figura de la izquierda) y del estadístico F_1 (figura de la derecha) para cada uno de los modelos de vectorización empleados en los *tweets* en castellano. En la Figura 14 se muestran los mismos resultados pero para los *tweets* en inglés.

Tabla 3: Tabla en la que se muestra el valor de las métricas accuracy y estadístico F_1 resultantes del estudio de la Task A. Se muestran los valores para cada uno de los modelos aplicados, para cada una de las representaciones de textos estudiadas y para los dos conjuntos de datos. En negrita se marcan los mejores resultados para cada caso.

		Castellano		Inglés	
		Accuracy	F1	Accuracy	F1
TD-IDF	Naive Bayes	0.656 ± 0.008	0.707 ± 0.008	0.730 ± 0.007	0.720 ± 0.007
	Regresión Logística	0.775 ± 0.009	0.769 ± 0.007	0.753 ± 0.006	0.730 ± 0.006
	SVM	0.775 ± 0.005	0.773 ± 0.005	0.755 ± 0.008	0.737 ± 0.008
	Random Forest	0.760 ± 0.008	0.745 ± 0.008	0.777 ± 0.007	0.766 ± 0.007
	Red neuronal	0.764 ± 0.004	0.766 ± 0.004	0.775 ± 0.008	0.752 ± 0.008
	Combinación de modelos	**0.786 ± 0.006**	**0.789 ± 0.006**	**0.779 ± 0.009**	**0.775 ± 0.009**
Bag of Words	Naive Bayes	0.625 ± 0.006	0.700 ± 0.006	0.723 ± 0.007	0.704 ± 0.007
	Regresión Logística	0.761 ± 0.008	0.765 ± 0.008	0.751 ± 0.007	0.735 ± 0.007
	SVM	0.760 ± 0.005	0.752 ± 0.005	0.758 ± 0.006	0.738 ± 0.006
	Random Forest	0.746 ± 0.008	0.746 ± 0.008	**0.762 ± 0.007**	**0.752 ± 0.007**
	Deep Learning	0.709 ± 0.007	0.699 ± 0.007	0.752 ± 0.008	0.754 ± 0.008
	Combinación de Modelos	**0.766 ± 0.004**	**0.773 ± 0.004**	0.754 ± 0.009	0.739 ± 0.009
Doc2Vec	Naive Bayes	0.613 ± 0.006	0.665 ± 0.006	0.641 ± 0.007	0.586 ± 0.007
	Regresión Logística	0.747 ± 0.006	0.748 ± 0.006	0.723 ± 0.009	0.707 ± 0.009
	SVM	**0.750 ± 0.006**	**0.752 ± 0.006**	**0.735 ± 0.009**	**0.719 ± 0.009**
	Random Forest	0.742 ± 0.008	0.735 ± 0.008	0.695 ± 0.008	0.684 ± 0.008
	Deep Learning	0.709 ± 0.007	0.699 ± 0.007	0.695 ± 0.006	0.694 ± 0.006
	Combinación de Modelos	0.743 ± 0.007	0.746 ± 0.007	0.707 ± 0.008	0.691 ± 0.008

Si se realiza una comparación entre los resultados, se puede ver que por lo general, el método de vectorización de textos que da un mejor resultado es TF-IDF. A pesar de esto, los resultados que se encuentran para el resto de métodos de vectorización son similares. Esto hace que las tres opciones sean buenas para vectorizar textos pequeños, como los *tweets*.

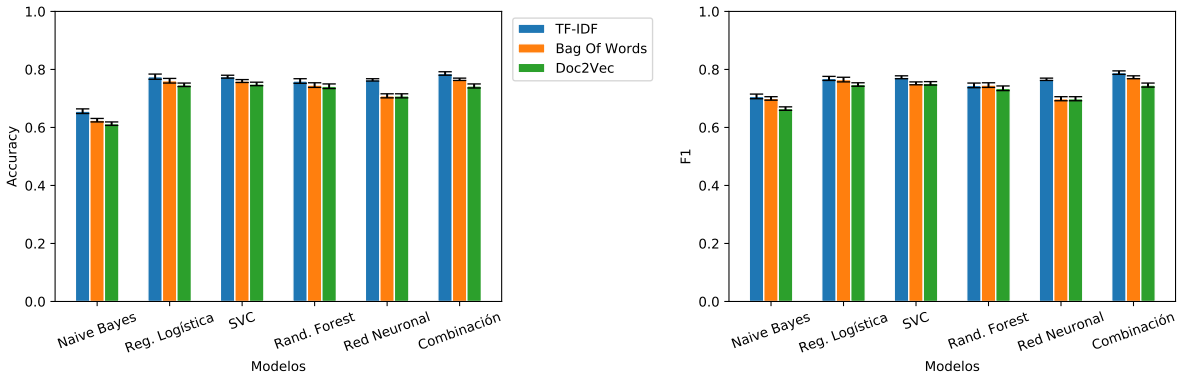

Figura 13: Gráficos de barras de los valores de accuracy (figura de la izquierda) y del estadístico F_1 (figura de la derecha) encontrados para el conjunto de datos en castellano.

En el conjunto de datos en castellano el modelo que ha obtenido mayor *accuracy* y mayor valor del estadístico F1 es el algoritmo de combinación de modelos ($accuracy = 0{,}786 \pm 0{,}006$, $F1 = 0{,}789 \pm 0{,}006$), seguido del *Support Vector Machine* ($accuracy = 0{,}775 \pm 0{,}005$, $F1 = 0{,}773 \pm 0{,}005$). En el conjunto de datos en inglés, el algoritmo de combinación de modelos vuelve a ser el que mejor realiza la clasificación según estas dos medidas ($accuracy = 0{,}779 \pm 0{,}009$, $F1 = 0{,}775 \pm 0{,}009$), seguido del modelo SVM ($accuracy = 0{,}777 \pm 0{,}007$, $F1 = 0{,}766 \pm 0{,}007$)

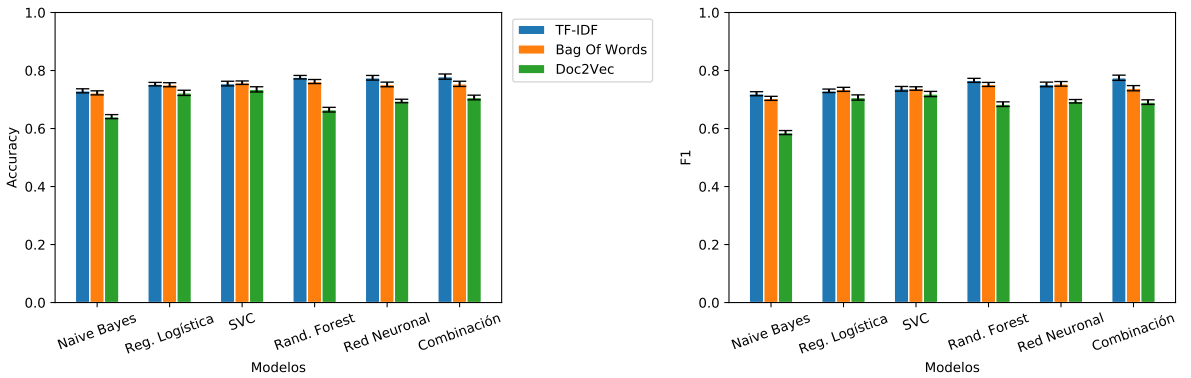

Figura 14: Gráficos de barras de los valores de accuracy (figura de la izquierda) y del estadístico F_1 (figura de la derecha) encontrados para el conjunto de datos en inglés.

Una vez se han visto los resultados numéricos, se pasa a analizar algunos de los casos de error que se han visto en el proceso de predicción. Este *tweet*, que se ve a continuación en el material anotado, está anotado como misógino. Sin embargo, a la hora de realizar la predicción se ha marcado como no misógino:

> « *En que se parece una mujer a un cientifico? En que al cientifico le pasan cosas por la cabeza, y a la mujer le pasan cabezas por la cosa* »

Este *tweet* no tiene ninguno de los elementos típicos como pueden ser insultos, uso de múltiples signos de exclamación o interrogación, *emojis*, ect. Esto hace más compleja la identificación automática de un lenguaje misógino. Un ejemplo del mismo caso en el conjunto de datos en inglés es el siguiente:

> « *Never trust a girl with 'makeup artist' in her bio... They're professional liars... They can even make your own eyes lie to you* »

A continuación se muestra la situación contraria: este *tweet* fue anotado como no misógino y sin embargo en la predicción se ha marcado como misógino:

> « *La perra les ladró a un grupo de adolescentes. Yo salí y le dije: 'Manu! callate! Dejá los gurises en paz!' Que vejez.* »

La razón de la mala clasificación es similar al caso anterior. En este mensaje aparece la palabra *perra*, que puede ser entendida automáticamente como un insulto aunque en este caso se hace referencia al animal. Con una representación semántica más precisa, la palabra "ladró" podría haber evitado el error.

Un ejemplo equivalente en el conjunto de datos en inglés es el siguiente.

> « *Being called a whore, slut, bitch and more for not 'playing along' with harassment #MeToo* »

Se trata un mensaje en el que se denuncian las palabras que reciben las mujeres. Al incluir estos términos de forma explícita, el modelo tiende a predecir el mensaje como misógino.

4.2 Clasificación de mensajes misóginos (*Task B*)

Una vez se ha conseguido tener un identificador que indica qué mensajes son misóginos y cuáles no, se pasa a la tarea de clasificación de los propios mensajes misóginos. En la sección 4.2.1 se ve la clasificación por objetivo, es decir, estudiar si el mensaje va dirigido a una persona en particular o a un grupo general. En la sección 4.2.2 se muestran los resultados para la clasificación de los *tweets* misóginos según las categorías vistas anteriormente.

Puesto que en la Task A se ha visto que el mejor modelo de vectorización de textos en este caso ha resultado ser la técnica TF-IDF, en el trabajo realizado para la Task B se ha usado solamente esta forma de vectorización de textos. Por este motivo no se hace una comparación entre esto.

4.2.1 Clasificación por objetivo

En la Tabla 4 se muestran los resultados de la métrica *accuracy* y de los valores del estadístico F_1 para cada uno de los modelos estudiados en el conjunto de datos en castellano. En la Tabla 5 se muestran los resultados correspondientes al conjunto de datos en inglés. Además, en la Figura 15 se muestra una representación gráfica de los resultados en castellano y en la Figura 16 para el conjunto de datos en inglés.

Tabla 4: Tabla en la que se muestra el valor de las métricas accuracy y estadístico F_1 resultantes del estudio de la tarea de clasificación según el objetivo de la Task B. Se muestran los valores para cada uno de los modelos aplicados con representación de textos TF-IDF para el conjunto de datos en castellano. En negrita se marcan los mejores resultados para cada caso.

	Castellano			
	Accuracy	F_{1micro}	F_{1macro}	F_{1media}
Naive Bayes	0.901 ± 0.006	0.901 ± 0.006	0.670 ± 0.006	0.881 ± 0.006
Regresión Logística	0.907 ± 0.006	0.907 ± 0.006	0.680 ± 0.006	0.886 ± 0.006
SVM	0.911 ± 0.005	0.911 ± 0.005	0.739 ± 0.005	0.901 ± 0.005
Random Forest	0.913 ± 0.006	0.913 ± 0.006	0.718 ± 0.006	0.897 ± 0.006
Red neuronal	**0.915 ± 0.009**	**0.915 ± 0.009**	**0.758 ± 0.009**	**0.903 ± 0.009**
Combinación de Modelos	0.911 ± 0.004	0.911 ± 0.004	0.713 ± 0.004	0.895 ± 0.004

En este caso, para el conjunto de datos el modelo de red neuronal es el modelo con el que se obtiene un mejor resultado ($accuracy = 0{,}915 \pm 0{,}009$, $F_{1macro} = 0{,}758 \pm 0{,}009$) seguido del modelo SVM ($accuracy = 0{,}911 \pm 0{,}005$, $F_{1macro} = 0{,}739 \pm 0{,}005$).

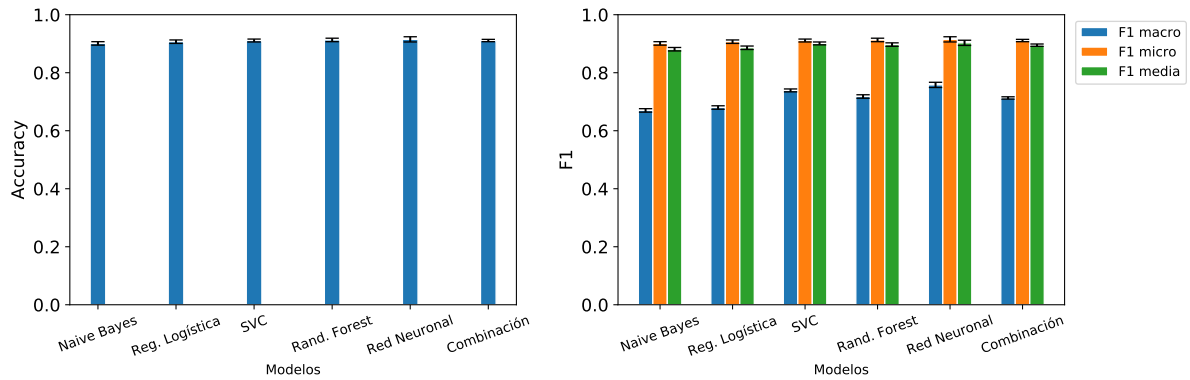

Figura 15: Gráficos de barras de los valores de accuracy y del estadístico F_{1macro}, F_{1micro} y F_{1media} vistos en la Tabla 4

Para el conjunto de datos en inglés, el mejor resultado se obtiene con el método Random Forest ($accuracy = 0{,}753 \pm 0{,}008$, $F_{1macro} = 0{,}719 \pm 0{,}008$) seguido del método de combinación de modelos ($accuracy = 0{,}742 \pm 0{,}006$, $F_{1macro} = 0{,}711 \pm 0{,}006$)

Tabla 5: Tabla en la que se muestra el valor de las métricas accuracy y estadístico F_1 resultantes del estudio de la tarea de clasificación según el objetivo de la Task B. Se muestran los valores para cada uno de los modelos aplicados con representación de textos TF-IDF para el conjunto de datos en inglés. En negrita se marcan los mejores resultados para cada caso.

	Inglés			
	Accuracy	F_{1micro}	F_{1macro}	F_{1avg}
Naive Bayes	0.702 ± 0.009	0.702 ± 0.009	0.656 ± 0.009	0.681 ± 0.009
Regresión Logística	0.742 ± 0.008	0.742 ± 0.008	0.709 ± 0.008	0.728 ± 0.008
SVM	0.743 ± 0.009	0.743 ± 0.009	0.708 ± 0.009	0.729 ± 0.009
Random Forest	**0.753 ± 0.008**	**0.753 ± 0.008**	**0.719 ± 0.008**	**0.738 ± 0.008**
Red neuronal	0.694 ± 0.008	0.694 ± 0.008	0.680 ± 0.008	0.692 ± 0.008
Combinación de Modelos	0.742 ± 0.006	0.742 ± 0.006	0.711 ± 0.006	0.730 ± 0.006

Como se puede ver, los resultados que se han obtenido tanto para el conjunto de mensajes en castellano como para el conjunto en inglés son realmente buenos. A pesar de esto, los resultados en castellano son algo mejores. Una posible causa de esta dife-

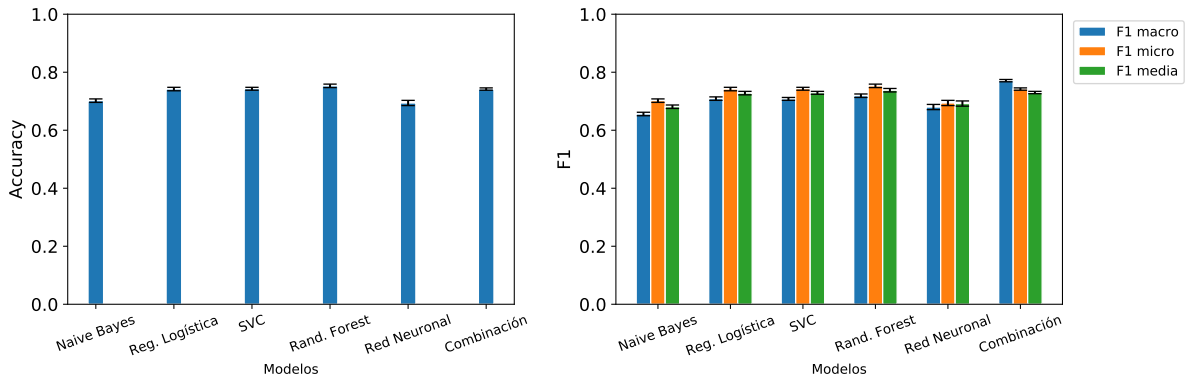

Figura 16: Gráficos de barras de los valores de accuracy y del estadístico F_{1macro}, F_{1micro} y F_{1media} vistos en la Tabla 5

rencia es en el conjunto de datos en castellano se enfatizase más a quién iba dirigido cada mensaje.

Un ejemplo en el que la clasificación se ha realizado de manera errónea es el siguiente:

> « *¿De qué murió Cenicienta? De que a las doce el tampax se le convirtió en calabaza* »

Inicialmente este *tweet* había sido marcado como pasivo, es decir, iba dedicado a un público general. En cambio la predicción realizada lo ha anotado como activo. En este caso no se ha tenido en cuenta el contexto general y al aparecer el nombre "Cenicienta" de forma explícita en el mensaje, se ha clasificado como un *tweet* dirigido a una persona real.

Con el siguiente ejemplo ocurre lo contrario. Este mensaje está anotado como activo, pues tiene como objetivo principal a Helen Keller. Sin embargo, en la clasificación se ha marcado como pasivo. Esto indica que el modelo no ha detectado a Helen Keller como el nombre de una persona. Aun así, según el contexto, este *tweet* puede ser considerado como general puesto que atribuye el hecho de no poder conducir a ser una mujer.

> « *Why couldn't Helen Keller drive? Because she is a woman* »

Un caso de disputa según el contexto también es el siguiente. Este mensaje está marcado como pasivo. Sin embargo en la clasificación se ha marcado como activo.

Depende de cómo se entienda, el mensaje puede ir dirigido a todas las mujeres o solo a las mujeres pelirrojas.

> « *Life is all about balance, so make sure you have as many blondes as you do brunettes on your roster... But no gingers, they're immoral* »

4.2.2 Clasificación por tema

En el caso del estudio de la clasificación de *tweets* misóginos según las categorías descritas anteriormente, en la Tabla 6 se muestran los resultados para el conjunto de datos en castellano junto con la Figura 17 en la que se muestra la representación mediante un gráfico de barras.

Tabla 6: Tabla en la que se muestra el valor de las métricas accuracy y estadístico F_1 resultantes del estudio de la tarea de clasificación según el tema de la Task B. Se muestran los valores para cada uno de los modelos aplicados con representación de textos TF-IDF para el conjunto de datos en castellano. En negrita se marcan los mejores resultados para cada caso.

	Castellano			
	Accuracy	F_{1micro}	F_{1macro}	F_{1avg}
Naive Bayes	0.614 ± 0.006	0.614 ± 0.006	0.274 ± 0.006	0.520 ± 0.006
Regresión Logística	0.664 ± 0.009	0.664 ± 0.009	0.399 ± 0.009	0.619 ± 0.009
SVM	$\mathbf{0.692 \pm 0.009}$	$\mathbf{0.692 \pm 0.009}$	$\mathbf{0.443 \pm 0.009}$	$\mathbf{0.651 \pm 0.009}$
Random Forest	0.681 ± 0.009	0.681 ± 0.009	0.442 ± 0.009	0.649 ± 0.009
Red neuronal	0.621 ± 0.007	0.561 ± 0.007	0.422 ± 0.006	0.597 ± 0.006
Combinación de Modelos	0.656 ± 0.009	0.656 ± 0.009	0.362 ± 0.009	0.593 ± 0.009

Para los mensajes en castellano, el modelo con el que se ha obtenido un mejor resultado es el algoritmo **Support Vector Machine** ($accuracy = 0{,}753 \pm 0{,}008$, $F_{1macro} = 0{,}719 \pm 0{,}008$) seguido del algoritmo Random Forest ($accuracy = 0{,}681 \pm 0{,}009$, $F_{1macro} = 0{,}442 \pm 0{,}009$) .

A continuación se muestran dos casos en los que la anotación de forma automática no coincide con la anotación del conjunto de test.

En el siguiente *tweet*, en el conjunto de test se había marcado este mensaje misógino de tipo *dominación*. Sin embargo, en la anotación automática se ha anotado como

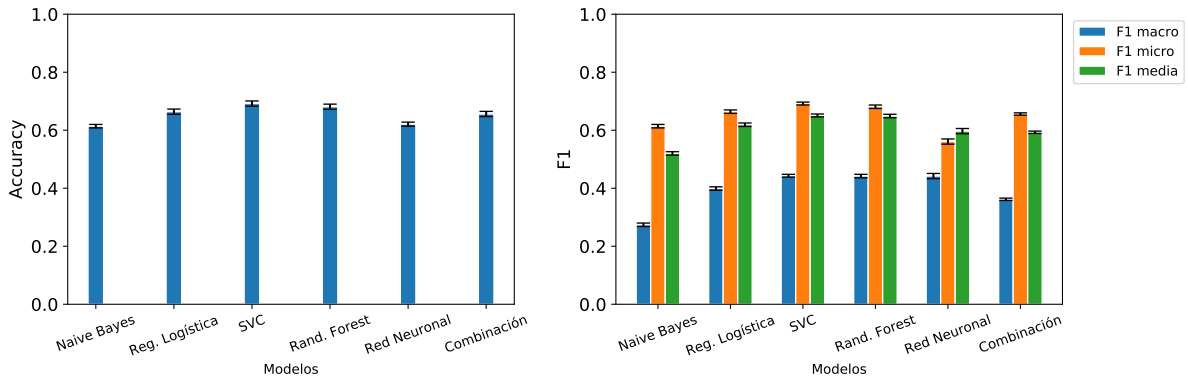

Figura 17: Gráficos de barras de los valores de accuracy y del estadístico F_{1macro}, F_{1micro} y F_{1media} vistos en la Tabla 6

descrédito. En este *tweet* está presente la palabra *perra*, por lo que el mensaje es de dominación haciendo referencia a la orden de callarse), pero al contener una palabra típicamente presente en *tweets* que tienen el objetivo de descrédito, se identifica como tal de forma automática. Como puede verse, es un caso sutil de diferenciar.

> « *@USER Cállate perra de Twitter* »

Por otro lado, el *tweet* que se ve a continuación estaba marcado en el conjunto test como *sexual harassment*. Sin embargo, en el proceso de clasificación automática se ha anotado como descrédito. Puesto que en el *tweet* aparece la palabra "puta" de forma repetida se ha tomado en la clasificación automática que el contexto del mensaje no estaba relacionado con un abuso sexual sino como una forma de insulto.

> « *@USER @USER Mmmmm la mas tetona y puta de Mexico, pinche puta tan buena . Mmmm te mereces k te usan, abusan en la cama, y te cogen como la puta k eres, mmmm mama una verga puta* »

Algo similar sucede con el siguiente caso. En el conjunto anotado este mensaje está clasificado como *derailing* y sin embargo, debido a la presencia de palabras consideradas como insulto como "zorra", la clasificación automática la toma como descrédito.

> « *Una mujer es tildada de zorra, buscafama etc y automáticamente está mintiendo, se le falta el respeto y se le cree al hombre. Cualquier semejanza con lo que consumimos todas las noches, es pura casualidad.* »

La mayoría de casos en los que la clasificación no se ajusta a lo que se tiene en el conjunto test es debido a casos como los que se han expuesto y casi todos estos casos han sido caracterizados dentro del tipo descrédito. Esto también es consecuencia de que el conjunto de datos no está bien balanceado y la presencia de *tweets* de descrédito es mucho mayor que del resto.

Si ahora se realiza el mismo estudio para los *tweets* en inglés, se obtienen los resultados que se muestran en la Tabla 7 junto con la representación gráfica de vista en la Figura 18.

Tabla 7: Tabla en la que se muestra el valor de las métricas accuracy y estadístico F_1 resultantes del estudio de la tarea de clasificación según el tema de la Task B. Se muestran los valores para cada uno de los modelos aplicados con representación de textos TF-IDF para el conjunto de datos en inglés. En negrita se marcan los mejores resultados para cada caso.

	Inglés			
	Accuracy	F_{1micro}	F_{1macro}	F_{1avg}
Naive Bayes	0.607 ± 0.005	0.607 ± 0.005	0.181 ± 0.005	0.492 ± 0.005
Regresión Logística	0.624 ± 0.008	0.624 ± 0.008	0.246 ± 0.008	0.567 ± 0.008
SVM	0.630 ± 0.005	0.630 ± 0.005	0.227 ± 0.005	0.554 ± 0.005
Random Forest	**0.635 ± 0.005**	**0.635 ± 0.005**	**0.258 ± 0.005**	**0.575 ± 0.005**
Red neuronal	0.620 ± 0.006	0.620 ± 0.006	0.157 ± 0.006	0.388 ± 0.006
Combinación de Modelos	0.633 ± 0.002	0.633 ± 0.002	0.219 ± 0.002	0.546 ± 0.002

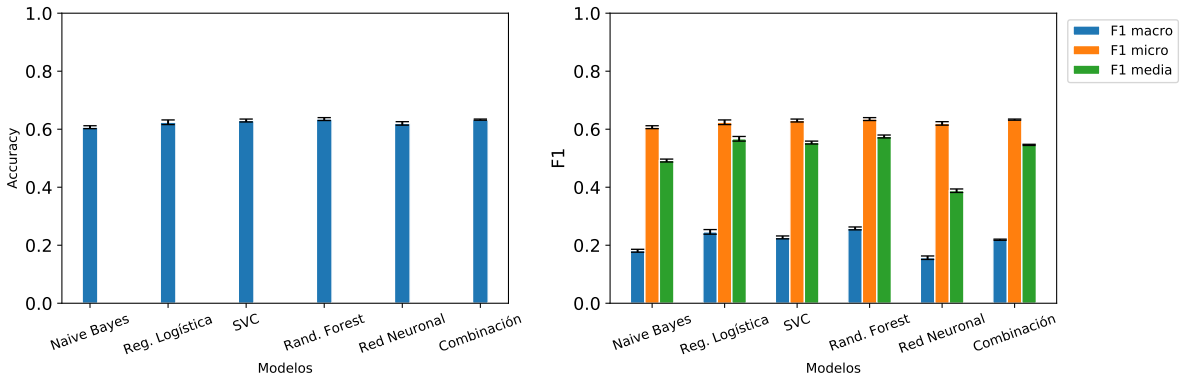

Figura 18: Gráficos de barras de los valores de accuracy y del estadístico F_{1macro}, F_{1micro} y F_{1media} vistos en la Tabla 7

Es sencillo ver que el modelo con el que se ha obtenido un mejor resultado es el algoritmo Random Forest ($accuracy = 0{,}635 \pm 0{,}005$, $F_{1macro} = 0{,}258 \pm 0{,}005$) segui-

do del algoritmo de combinación de modelos (*accuracy* $= 0{,}633 \pm 0{,}002$, $F_{1macro} = 0{,}219 \pm 0{,}002$)

Ahora, al igual que antes, se muestra una serie de ejemplos de clasificaciones erróneas para el conjunto de datos en inglés.

El siguiente mensaje estaba anotado en el conjunto test como *sexual harassment*. Sin embargo, en la clasificación automática se ha marcado como descrédito.

> « *I'm back in it bitch If the head game is right and that pussy feel good Throw a rack at that bitch* »

Al aparecer la palabra "bitch" dos veces, la clasificación automática considera que estos dos insultos tienen más peso que el resto del contexto y por esto se clasifica como descrédito.

Un ejemplo del caso contrario es el siguiente:

> « *An angry woman standing up4 human rights is 2 difficult to be worth loving. Men love pretty slave dolls who keep their mouth shut or on dick* »

Este *tweet* estaba anotado como descrédito en el conjunto test, pero al contener varias palabras que suelen aparecer en *tweets* con contenido sexual como "mouth" o "dick" se ha denotado de forma automática como *sexual harassment*.

Por último, este *tweet* en el conjunto test estaba anotado como estereotipo, pero en la clasificación automática ha sido categorizado como descrédito:

> « *RT @JWMofficial: If you play field hockey there's a good chance you're a lesbian.* »

El clasificador automático no reconoce el contexto de la palabra "lesbian" y lo interpreta como un insulto cuando en este caso está siendo utilizada como forma de generalizar el comportamiento de un colectivo.

4.3 Comparación de resultados

Después de realizar todo el estudio, los mejor resultados que se han obtenido a lo largo del trabajo son los que se muestran en la Tabla 8. En un primer análisis se puede ver que de los dos conjuntos de datos, el que tiene los *tweets* en castellano es más sencillo de caracterizar que el conjunto de datos en inglés puesto que las métricas obtenidas al aplicar los diferentes métodos son algo mejores.

Tabla 8: Resumen de los mejores resultados encontrados para las métricas accuracy y del estadístico F_{1macro}

		Castellano		Inglés	
		Accuracy	F_{1macro}	Accuracy	F_{1macro}
Task A		0.786 ± 0.006	0.789 ± 0.006	0.779 ± 0.009	0.75 ± 0.009
Task B	Objetivo	0.915 ± 0.009	0.758 ± 0.009	0.753 ± 0.008	0.719 ± 0.008
	Tema	0.692 ± 0.009	0.443 ± 0.009	0.635 ± 0.005	0.258 ± 0.005
	Media		0.601 ± 0.009		0.489 ± 0.009

Si se fija la atención en las diferencias que existen entre los problemas Task A y Task B, se encuentran resultados algo más variados. En el caso de la Task A, la métrica *accuracy* está alrededor del 0.8, es decir, alrededor del 80 % de las predicciones que se han realizado han sido correctas. Esto se puede considerar como un buen resultado. En el estudio de Task B se ve una gran diferencia entre la clasificación según el objetivo y según el tema. Según el objetivo se obtiene un $accuracy = 0{,}927$ para el conjunto de datos en castellano, que es un muy buen resultado. En cambio, para el estudio del tema del mensaje misógino se tiene $accuracy = 0{,}697$, que no denota tanto que la clasificación no es tan buena sino que la métrica *accuracy* no aporta mucha información debido a que las clases no están correctamente balanceadas.

De forma similar, se pueden observar las diferencias a través de los resultados del estadístico F_{1macro}. Esta diferencia dentro de los estudios de la Task B son debido al carácter del estudio. En el caso del objetivo, la clasificación es binaria y por tanto el aprendizaje del modelo es más sencilla. En cambio, cada *tweet* puede ser clasificado en cinco temas diferentes, que sumado a la ambigüedad del lenguaje y más en un entorno

informal como las redes sociales, resulta un trabajo mucho más complejo.

Además, como se ha visto anteriormente, las clases no están del todo bien balanceadas y predomina la presencia de *tweets* de descrédito. Esto hace que aunque se tomen medidas y herramientas para poder mitigar los posibles errores, a la hora de realizar el entrenamiento se necesitan más referencias para poder caracterizar el resto de tipos.

Ahora bien, realizar un estudio absoluto de los resultados, sin tener una referencia con la que poder comparar no permite saber si el resultado que se ha encontrado es bueno o no. Por tanto, se van a comparar los resultados encontrados con los que encontraron otros participantes del reto IberVal2018. Estos resultados se pueden ver en la Tabla 9. En dicha tabla se muestran los valores de la métrica *accuaracy* para la Task A, el valor del estadístico F_{1macro} para los dos estudios de la Task B además la media aritmética de ambas (según objetivo y tema), que es una métrica que se proponen en la competición para tener un resultado global de la tarea.

No todos los participantes realizaron todos los estudios. De hecho el estudio con el conjunto de datos en castellano fue menos trabajado en comparación con el de inglés. Por tanto, la comparación se realizará respecto a los valores que se tienen. Además en la competición se da un valor de referencia (AMI baseline) con el que poder tener una primera aproximación.

En las últimas columnas también se indican los métodos de representación de textos que han utilizado los participantes además de los algoritmos de clasificación utilizados. Esto permite también realizar una comparación de los métodos que se han utilizado en este trabajo.

Respecto a la Task A, el valor de accuracy que se ha obtenido en este trabajo para el conjunto de datos en castellano ($accuracy = 0{,}786 \pm 0{,}006$) es algo mayor al del AMI baseline ($accuracy = 0{,}768$), pero se ve superado por el resto de participantes. El valor encontrado no es demasiado malo, pero hay otros modelos con los que se obtiene un mejor resultado. Algo similar pasa con el conjunto de datos en inglés. El valor encontrado en este trabajo es $accuracy = 0{,}779 \pm 0{,}009$ y está por algo por debajo del valor de referencia ($accuracy = 0{,}783$). Sin embargo se puede ver que otros participantes como JoseSebastian, ITT o GrCML2016 también están por debajo. El valor de este trabajo se encuentra dentro de los valores recogidos por otros participantes.

Tabla 9: Recopilación de los valores de la métrica accuracy para la Task A y del estadístico F_{1macro} para la Task B de algunos participantes de la competición AMI.
Los resultados de JoseSebastian se muestran en [Canós, 2018]), los de 14-ExLab en [Pamungkas et al., 2018], los de resham en [Ahluwalia et al., 2018], los de ixaTeam en [Goenaga et al., 2018], los de ITT en [Shushkevich and Cardiff, 2018], los de SB en ([Frenda et al., 2018]), los de GrCML2016 en [Liu et al., 2018] y los de _vic_ en [Nina-Alcocer, 2018].
También se añade el valor de referencia provisto por la organización de la competición AMI ([Fersini et al., 2018])

	Castellano				Inglés				Representación	Algortimo
	Task A	Task B			Task A	Task B				
		Objetivo	Tema	Media		Objetivo	Tema	Media		
JoseSebastian	0.815	0.542	0.323	0.433	0.749	0.505	0.148	0.326	TF-IDF	SVM
14-ExLab	0.815	0.553	0.339	0.446	0.913	0.581	0.158	0.369	BoW	SVM
resham	-	-	-	-	0.785	0.555	0.148	0.351	BoW	Combinación + RNN
ixaTeam	0.796	-	-	-	0.789	-	-	-	BoW	RNN + Bi-LSTM + CRF
ITT	-	-	-	-	0.759	0.457	0.180	0.318	TF-IDF	Combinación
SB	0.813	0.552	0.330	0.441	0.871	0.592	0.292	0.442	TF-IDF	SVM
GrCML2016	-	-	-	-	0.528	0.270	0.086	0.178	Doc2Vec	Combinación
vic	0.805	0.534	0.320	0.427	0.781	0.541	0.138	0.340	TF-IDF	Combinación
AMI baseline	0.768	0.537	0.281	0.409	0.783	0.518	0.157	0.337		

Resumen de abreviaciones.
SVM (Suport Vector Machine), BoW (Bag Of Words), RNN (Recurrent Neural Network), Bi-LSTM (Bidirectional Long Short Term Memory), CRF (Condi- tional Random Fields)

Tanto para el caso del conjunto de datos en castellano como para el conjunto de datos en inglés, el método de vectorización utilizado ha sido TF-IDF con un algoritmo de aprendizaje automático de combinación de modelos. Teniendo esto en cuenta, los resultados pueden ser comparados con el equipo ITT y con _vic_ , que han usado la misma combinación de métodos.

En la Figura 19 se muestra un diagrama de caja en el que de forma gráfica se puede comparar los resultados conseguidos por los equipos participantes con el valor de referencia de AMI y con el valor encontrado en este trabajo.

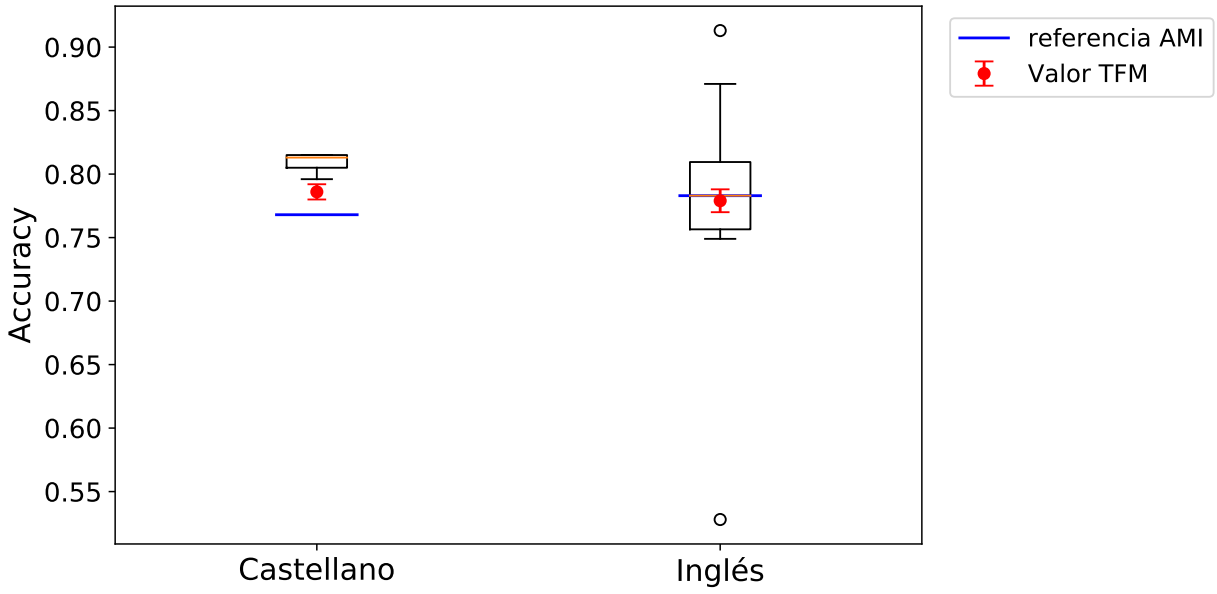

Figura 19: Diagrama de cajas de los valores del estadístico F_{1macro} del conjunto de participantes encontrados para la Tarea A. Con una línea azul se muestra el valor de referencia y con un punto rojo el mejor resultado encontrado en este trabajo.

Respecto a la Task B, si se comparan los resultados obtenidos con los obtenidos por el resto de participantes, se puede observar una cierta mejora. En el caso de la clasificación según el objetivo, los mejores resultados se han obtenido con un algoritmo de Red Neuronal y un algoritmo de Random Forest para el conjunto de datos en castellano e inglés respectivamente. En la gráfica de arriba a la izquierda de la Figura 20 se muestra una comparación para esta tarea de clasificación por objetivos para ambas lenguas. Se ve que en ambos casos se ha obtenido un mejor resultado que el resto de participantes y que el valor de referencia de la competición.

Si se centra ahora la atención en la clasificación según el tema, el resultado que se obtiene es similar. En este caso los mejores resultados corresponden a un algoritmo de *Support Vector Machine* y a un algoritmo Random Forest para el conjunto de datos en castellano e inglés respectivamente. Para el conjunto en castellano se obtiene el mejor valor de la métrica F_{1macro} respecto al resto de participantes y para el conjunto de datos en inglés se obtiene el segundo mejor valor. Esta comparación se puede observar en la gráfica de arriba a la derecha de la Figura 20.

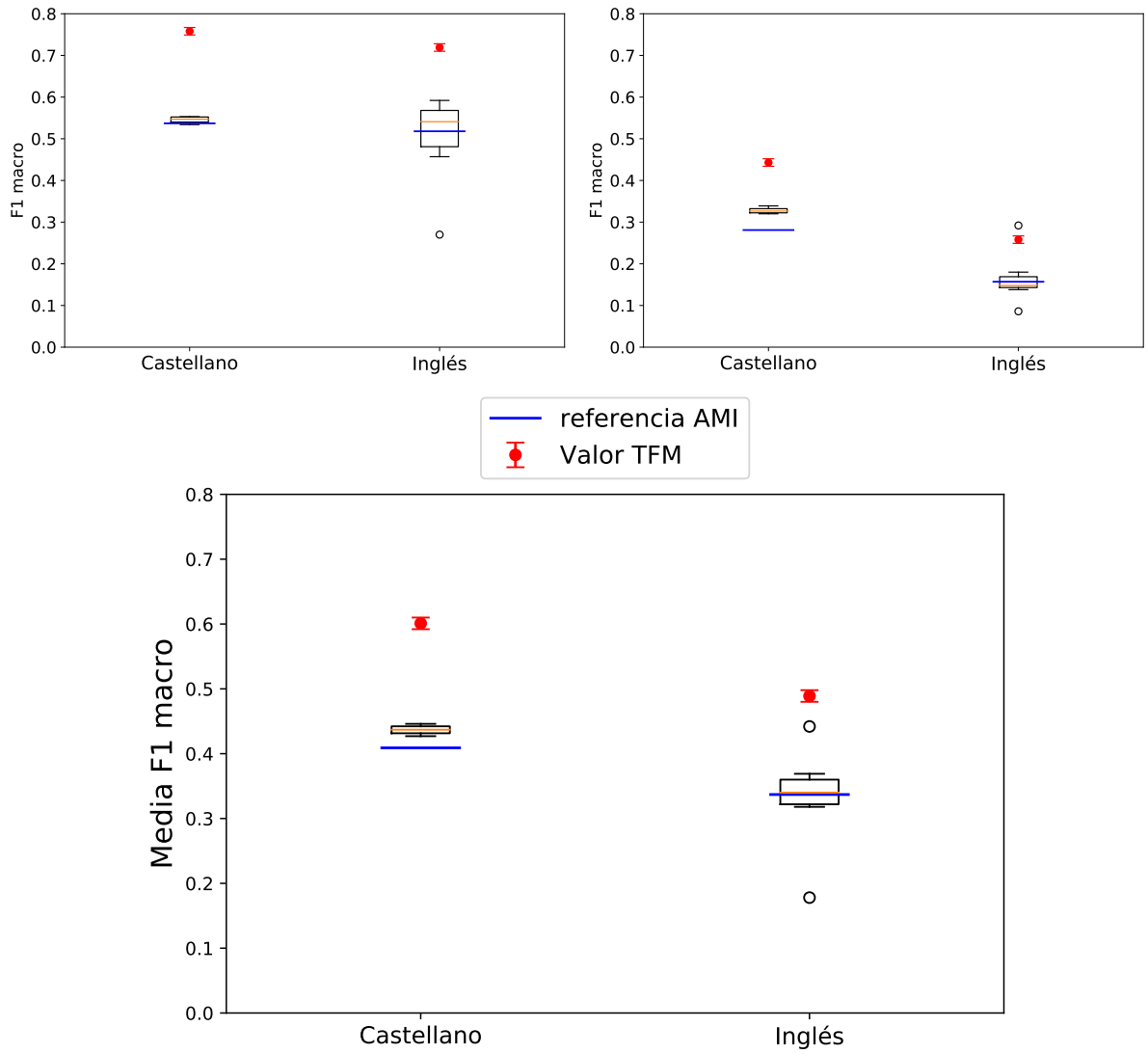

Figura 20: Diagramas de cajas de los valores de la métrica accuracy del conjunto de participantes encontrados para los dos casos de la Tarea B. En la línea de arriba se muestra el resultado para la clasificación según el objetivo a la izquierda y el resultado para la clasificación según el tema a la derecha. La figura de abajo es el resultado para la media de los dos resultados anteriores. Con una línea azul se muestra el valor de referencia y con un punto rojo el mejor resultado encontrado en este trabajo.

Finalmente en la gráfica de abajo de la Figura 20 se muestra la comparación con el valor medio de las dos clasificaciones de la Task B. En este caso, la media de los valores encontrados en este trabajo para el estadístico F_{1macro} es mayor que la encontrada tanto por el resto de participantes como por el valor de referencia de la competición.

Capítulo 5

Conclusiones

Finalmente, en este último apartado se van a dar unas conclusiones acerca de los resultados que se han obtenido y del conjunto del trabajo.

Lo primero que se ha hecho es realizar una tarea de preprocesamiento de los datos muy necesaria porque como se ha visto, los datos que vienen de redes sociales, como *Twitter*, tienen un gran componente de ruido. Una vez hecho este proceso se ha pasado al entrenamiento de los diversos modelos. Para ello se ha de realizar primero un proceso de vectorización de los textos. Se ha concluido que aunque los tres métodos usados (*TF-IDF*, *Bag Of Words* y *Doc2Vec*) no guardan grandes diferencias entre sí, el mejor modelo de vectorización para este conjunto de datos es *TF-IDF*. Por este motivo en la Task B únicamente se usará este método.

A continuación se pasa al estudio de los modelos de entrenamiento. Se ha visto que los mejores resultados que se han encontrado en la tarea de identificación de mensajes misóginos (Task A) son para el método de aprendizaje automático de Combinación de Modelos. Con el conjunto de datos en castellano se ha conseguido $accuracy = 0{,}786 \pm 0{,}006$ y $F_1 = 0{,}789 \pm 0{,}006$. Con el conjunto de datos en inglés $accuracy = 0{,}779 \pm 0{,}009$ y $F_1 = 0{,}775 \pm 0{,}009$

Los resultados encontrados no son objetivamente malos, pues alrededor del 78 % de los *tweets* estudiados se han clasificado adecuadamente en ambos casos. En el caso del conjunto de datos en castellano se mejora el valor de referencia dado y en el caso del conjunto de datos en inglés queda ligeramente por debajo, pero se ve cubierto por el error. Sin embargo, se ha visto que en el resto de equipos de la competición han

aparecido otros modelos que mejoran notablemente la predicción.

Sin embargo, esta tendencia cambia en el estudio de la Task B. Para la subtarea de clasificación por objetivo para el conjunto de datos en castellano, los mejores resultados se han obtenido con un algoritmo de red neuronal ($accuracy = 0{,}915 \pm 0{,}009$ y $F_{1macro} = 0{,}758 \pm 0{,}009$). Para el conjunto de datos en inglés, los mejores resultados se han obtenido con un algoritmo Random Forest ($accuracy = 0{,}753 \pm 0{,}008$ y $F_{1macro} = 0{,}719 \pm 0{,}008$). Esto supone un aumento de un 41.1 % y 37.1 % respectivamente sobre el valor de referencia de AMI y de un 38.8 % y 21.5 % respectivamente respecto del valor máximo de los participantes.

Para la subtarea de clasificación por tema, para el conjunto de datos en inglés, los mejores resultados se han obtenido con un algoritmo de Support Vector Machine ($accuracy = 0{,}692 \pm 0{,}009$ y $F_{1macro} = 0{,}443 \pm 0{,}009$). Para el conjunto de datos en inglés, los mejores resultados se han obtenido con un algoritmo Random Forest ($accuracy = 0{,}635 \pm 0{,}005$ y $F_{1macro} = 0{,}258 \pm 0{,}005$). Esto supone un aumento de un 57.7 % y 64.3 % respectivamente sobre el valor de referencia de AMI. Respecto del valor máximo encontrado por el resto de equipos, para los *tweets* en español el resultado se mejora en un 30.1 %. Para los *tweets* en inglés hay un equipo que ha obtenido un mejor resultado ($F_{1macro} = 0{,}292$).

En consecuencia a todos estos resultados, se puede concluir que los modelos llevados a cabo son en general buenos clasificadores. Por un lado, aunque existan otros modelos que identifiquen mejor los mensajes con contenido misógino, el resultado encontrado por sí mismo es bastante bueno. Sin embargo, se ha encontrado un modelo muy bueno tanto para identificar el tipo de objetivo al que va dirigido un *tweet* y el tipo de misoginia del que está haciendo uso.

Como elemento de mejora para un futuro trabajo, se podría profundizar en el hecho del balanceo de los datos. En el caso de la clasificación por tema, la mayoría de los *tweets* pertenecían a la categoría descrédito. Esto hace que a la hora de realizar el entrenamiento haya una sobrerepresentación de este tipo y una subrepresentación del resto. Se han aplicado técnicas para reducir los problemas causados por este hecho, pero la diferencia es tan grande que las consecuencias son inevitables.

También se podría explorar más los modelos basados en redes neuronales (*Deep*

Learning). Se ha visto que en algún caso el resultado obtenido ha sido bastante bueno y sería interesante explorar más estos modelos, al igual que lo hacen algunos participantes, para intentar obtener mejores resultados.

Otro aspecto a estudiar es el uso de ironía en los mensajes. Es muy usual que en el mundo de las redes sociales, como expansión de la comunicación en la *vida offline*, las personas hagan uso de recursos como el sarcasmo o la ironía. A la hora de procesar estos textos, es muy probable que lleven a errores por no ser capaz de reconocer el contexto en el que se está enunciando un mensaje.

El estudio de casos de misoginia en la sociedad actual, al igual que el estudio de otros casos de odio como el racismo o la LGTBI-fobia son muy necesarios debido al aumento de casos que se está viendo en los últimos tiempos. Según fuentes de Moncloa, hasta el momento en 2021, el número de mujeres asesinadas a consecuencia de violencia machista asciende a 18 [Ministerio de Igualdad, 2021]. Según el Ministerio del Interior en su último informe acerca de los delitos de odio en 2019 [Ministerio del Interior, 2019], se registraron 1598 casos de este tipo de delito a lo largo del año. Un 8,3 % más que en el año anterior. En concreto las denuncias realizadas por casos de orientación sexual e identidad de género aumentaron un 8.6 % mientras que las denuncias por causa de racismo o xenofobia creció un 20.9 %

En los últimos años han aparecido diversos productos culturales en diversos ámbitos (como el álbum *PUTA* [Zahara, 2021] o la obra de teatro '*Juguetes Rotos* [Román, 2018]) que tienen como objetivo visibilizar este tipo de problemas y permitir así una reflexión por parte de la sociedad. Los estudios en el campo de la ciencia y la tecnología tienen que ser otro punto de apoyo que ayuden a dar respuesta a este tipo de problemas sociales. Este trabajo, bajo el campo de la ciencia de datos, pretende ser un paso más para que este tipo de agresiones cesen.

Bibliografía

[Ahluwalia et al., 2018] Ahluwalia, R., Shcherbinina, E., Callow, E., Nascimento, A. C., and De Cock, M. (2018). Detecting misogynous tweets. In *IberEval@ SEPLN*, pages 242–248. (document), 9

[Bisong, 2019] Bisong, E. (2019). Tensorflow 2.0 and keras. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pages 347–399. Springer. 3.3

[Braithwaite, 2014] Braithwaite, A. (2014). 'seriously, get out': Feminists on the forums and the war (craft) on women. *New media & society*, 16(5):703–718. 1.1

[Brownlee, 2017] Brownlee, J. (2017). *Deep learning for natural language processing: develop deep learning models for your natural language problems*, chapter Chapter 8. The Bag-of-Words Model. Machine Learning Mastery. 2.4.1

[Buitinck et al., 2013] Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., and Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122. 3.2, 3.3

[Cambria et al., 2017] Cambria, E., Poria, S., Gelbukh, A., and Thelwall, M. (2017). Sentiment analysis is a big suitcase. *IEEE Intelligent Systems*, 32(6):74–80. 1.1

[Canós, 2018] Canós, J. S. (2018). Misogyny identification through svm at ibereval 2018. In *IberEval@ SEPLN*, pages 229–233. (document), 9

[Chicco and Jurman, 2020] Chicco, D. and Jurman, G. (2020). The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13. 3.4.1, 3.4.1, 3.4.1, 3.4.1

[Christian et al., 2016] Christian, H., Agus, M. P., and Suhartono, D. (2016). Single document automatic text summarization using term frequency-inverse document frequency (tf-idf). *ComTech: Computer, Mathematics and Engineering Applications*, 7(4):285–294. 2.4.2

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297. (document), 7, 2.5.3

[Cutler et al., 2012] Cutler, A., Cutler, D. R., and Stevens, J. R. (2012). Random forests. In *Ensemble machine learning*, pages 157–175. Springer. 2.5.4, 2.5.4, 2.5.4, 2.5.4

[Duggan, 2017] Duggan, M. (2017). Online harassment 2017. 1.1

[Fayyad et al., 2003] Fayyad, U. M., Piatetsky-Shapiro, G., and Uthurusamy, R. (2003). Summary from the kdd-03 panel: Data mining: The next 10 years. *SIGKDD Explor. Newsl.*, 5(2):191?196. 2.3

[Fersini et al., 2018] Fersini, E., Rosso, P., and Anzovino, M. (2018). Overview of the task on automatic misogyny identification at ibereval 2018. *IberEval@ SEPLN*, 2150:214–228. (document), 1.1, 1.2, 1.4, 2.1, 2.1, 3.4.1, 9

[Frenda et al., 2018] Frenda, S., Bilal, G., et al. (2018). Exploration of misogyny in spanish and english tweets. In *Third Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018)*, volume 2150, pages 260–267. Ceur Workshop Proceedings. (document), 9

[Gaur et al., 2019] Gaur, M., Alambo, A., Sain, J. P., Kursuncu, U., Thirunarayan, K., Kavuluru, R., Sheth, A., Welton, R., and Pathak, J. (2019). Knowledge-aware assessment of severity of suicide risk for early intervention. In *The World Wide Web Conference*, WWW '19, page 514–525, New York, NY, USA. Association for Computing Machinery. 1.1

[Géron, 2019] Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*, chapter Chapter 7. Ensemble Learning and Random Forests. O'Reilly Media. 2.5.6

[Goenaga et al., 2018] Goenaga, I., Atutxa, A., Gojenola, K., Casillas, A., de Ilarraza, A. D., Ezeiza, N., Oronoz, M., Pérez, A., and Perez-de Viñaspre, O. (2018). Automatic misogyny identification using neural networks. In *IberEval@ SEPLN*, pages 249–254. (document), 9

[Haddi et al., 2013] Haddi, E., Liu, X., and Shi, Y. (2013). The role of text preprocessing in sentiment analysis. *Procedia Computer Science*, 17:26–32. 2.3

[Hewitt et al., 2016] Hewitt, S., Tiropanis, T., and Bokhove, C. (2016). The problem of identifying misogynist language on twitter (and other online social spaces). In *Proceedings of the 8th ACM Conference on Web Science*, pages 333–335. 1.1

[Honnibal and Montani, 2017] Honnibal, M. and Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear. 3.1

[Jakkula, 2006] Jakkula, V. (2006). Tutorial on support vector machine (svm). *School of EECS, Washington State University*, 37. 2.5.3

[Joachims, 1996] Joachims, T. (1996). A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, Carnegie-mellon univ pittsburgh pa dept of computer science. 2.4.2

[Kim et al., 2019] Kim, D., Seo, D., Cho, S., and Kang, P. (2019). Multi-co-training for document classification using various document representations: Tf–idf, lda, and doc2vec. *Information Sciences*, 477:15–29. (document), 2.4.3, 3, 2.4.3

[Kleinbaum et al., 2002] Kleinbaum, D. G., Dietz, K., Gail, M., Klein, M., and Klein, M. (2002). *Logistic regression*. Springer. (document), 2.5.2, 2.5.2, 6

[Le and Mikolov, 2014] Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR. 2.4.3

[Liu et al., 2018] Liu, H., Chiroma, F., and Cocea, M. (2018). Identification and classification of misogynous tweets using multi-classifier fusion. In *Evaluation of Human Language Technologies for Iberian Languages: IberEval 2018*, pages 268–273. CEUR Workshop Proceedings. (document), 9

[Loper and Bird, 2002] Loper, E. and Bird, S. (2002). Nltk: The natural language toolkit. *arXiv preprint cs/0205028*. 3.1

[Malpas, 2008] Malpas, J. (2008). The non-autonomy of the virtual: Philosophical reflections on contemporary virtuality. 1.1

[Martins et al., 2003] Martins, C. A., Monard, M. C., and Matsubara, E. T. (2003). Reducing the dimensionality of bag-of-words text representation used by learning algorithms. In *Proc of 3rd IASTED International Conference on Artificial Intelligence and Applications*, pages 228–233. 2.4.1, 2.4.1

[Mikolov et al., 2013] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*. 2.4.3

[Mohandes et al., 2018] Mohandes, M., Deriche, M., and Aliyu, S. O. (2018). Classifiers combination techniques: A comprehensive review. *IEEE Access*, 6:19626–19639. (document), 12, 2.5.6, 2.5.6

[Mueller, 2020] Mueller, A. (2020). Wordcloud. http://amueller.github.io/word_cloud. 3.1

[Nina-Alcocer, 2018] Nina-Alcocer, V. (2018). Ami at ibereval2018 automatic misogyny identification in spanish and english tweets. In *IberEval@ SEPLN*, pages 274–279. (document), 9

[Noble, 2006] Noble, W. S. (2006). What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567. 2.5.3

[Pamungkas et al., 2020] Pamungkas, E. W., Basile, V., and Patti, V. (2020). Misogyny detection in twitter: a multilingual and cross-domain study. *Information Processing & Management*, 57(6):102360. 1.1, 1.2

[Pamungkas et al., 2018] Pamungkas, E. W., Cignarella, A. T., Basile, V., Patti, V., et al. (2018). 14-exlab@ unito for ami at ibereval2018: Exploiting lexical knowledge for detecting misogyny in english and spanish tweets. In *3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages, IberEval 2018*, volume 2150, pages 234–241. CEUR-WS. (document), 9

[Pang et al., 2020] Pang, B., Nijkamp, E., and Wu, Y. N. (2020). Deep learning with tensorflow: A review. *Journal of Educational and Behavioral Statistics*, 45(2):227–248. 2.5.5, 2.5.5, 2.5.5

[Psomakelis et al., 2015] Psomakelis, E., Tserpes, K., Anagnostopoulos, D., and Varvarigou, T. (2015). Comparing methods for twitter sentiment analysis. *arXiv preprint arXiv:1505.02973*. 2.3

[Ramasubramanian and Singh, 2017] Ramasubramanian, K. and Singh, A. (2017). *Machine learning using R*, chapter Chapter 6.11 Artificial Neural Networks. Number 1. Springer. (document), 2.5.5, 2.5.5, 2.5.5, 9, 10, 11

[Román, 2018] Román, C. (2018). Juguetes rotos. Producciones Rocambolescas. Con Kike Guaza y Nacho Guerreros, http://carolinaroman.es/portfolio/juguetes-rotos/. 5

[Rong, 2014] Rong, X. (2014). word2vec parameter learning explained. *arXiv preprint arXiv:1411.2738*. (document), 4

[Salton, 1989] Salton, G. (1989). *Automatic text processing: The transformation, analysis, and retrieval of Information by Computer*, volume 169, chapter Chapter 9. Automatic Indexing. Addison-Wesley Longman Publishing Co., Inc. 2.4.2

[Salton and Buckley, 1988] Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523. 2.4.2

[Schütze et al., 2008] Schütze, H., Manning, C. D., and Raghavan, P. (2008). *Introduction to information retrieval*, volume 39, chapter Chapter 6. Scoring, term weighting,

and the vector space model. Cambridge University Press Cambridge. 2.4.2, 2.4.2, 2.4.2, 2.4.2

[scikit-learn developers, 2020a] scikit-learn developers (2020a). Combinación de modelos. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.VotingClassifier.html. 3.3

[scikit-learn developers, 2020b] scikit-learn developers (2020b). Gaussian nb. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html. 3.3

[scikit-learn developers, 2020c] scikit-learn developers (2020c). Logistic regression. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html. 3.3

[scikit-learn developers, 2020d] scikit-learn developers (2020d). Random forest. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. 3.3

[scikit-learn developers, 2020e] scikit-learn developers (2020e). Svc. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html. 3.3

[Shanmugamani, 2018] Shanmugamani, R. (2018). *Deep Learning for Computer Vision: Expert techniques to train advanced neural networks using TensorFlow and Keras*, chapter Chapter 1. Getting Started. Packt Publishing Ltd. (document), 2.5.5, 8, 2.5.5, 2.5.5, 2.5.5.1, 2.5.5.2, 2.5.5.2

[Shushkevich and Cardiff, 2018] Shushkevich, E. and Cardiff, J. (2018). Classifying misogynistic tweets using a blended model: The ami shared task in ibereval 2018. In *IberEval@ SEPLN*, pages 255–259. (document), 9

[Solomon, 2020] Solomon, B. (2020). Demoji. https://pypi.org/project/demoji. 3.1

[spaCy, 2021] spaCy (2021). spacy 101: Everything you need to know. https://spacy.io/usage/spacy-101. 2.3

[Symeonidis et al., 2018] Symeonidis, S., Effrosynidis, D., and Arampatzis, A. (2018). A comparative evaluation of pre-processing techniques and their interactions for twitter sentiment analysis. *Expert Systems with Applications*, 110:298–310. 2.3

[Taheri and Mammadov, 2013] Taheri, S. and Mammadov, M. (2013). Learning the naive bayes classifier with optimization models. *International Journal of Applied Mathematics and Computer Science*, 23(4). (document), 2.5.1, 2.5.1, 5, 2.5.1

[Ministerio de Igualdad, 2021] Ministerio de Igualdad (2021). Igualdad condena un nuevo asesinato por violencia de género en Sevilla. *Moncloa*. https://www.lamoncloa.gob.es/serviciosdeprensa/notasprensa/igualdad/Paginas/2021/140621-asesinato_genero_sevilla.aspx. 5

[Ministerio del Interior, 2019] Ministerio del Interior (2019). Informe de la Evolución de los Delitos de Odio del Ministerio del Interior del año 2019. *Ministerio del Interior*. 5

[Tokunaga and Makoto, 1994] Tokunaga, T. and Makoto, I. (1994). Text categorization based on weighted inverse document frequency. In *Special Interest Groups and Information Process Society of Japan (SIG-IPSJ*. Citeseer. 2.4.2, 2.4.2, 2.4.2

[Tsoumakas and Katakis, 2007] Tsoumakas, G. and Katakis, I. (2007). Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3):1–13. 3.4.2

[VanDerWerff, 2014] VanDerWerff, E. (2014). #gamergate: Here's why everybody in the video game world is fighting. *Vox.com*. https://www.vox.com/2014/9/6/6111065/gamergate-explained-everybody-fighting. 2

[Vani and Rao, 2019] Vani, S. and Rao, T. M. (2019). An experimental approach towards the performance assessment of various optimizers on convolutional neural network. In *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, pages 331–336. IEEE. 3.4.2, 3.4.2, 3.4.2

[Vapnik, 1963] Vapnik, V. (1963). Pattern recognition using generalized portrait method. *Automation and remote control*, 24:774–780. 2.5.3

[Vapnik, 2006] Vapnik, V. (2006). *Estimation of dependences based on empirical data*. Springer Science & Business Media. 2.5.3

[Wang, 2005] Wang, L. (2005). *Support vector machines: theory and applications*, volume 177, chapter Chapter 1. Support Vector Machines ? An Introduction. Springer Science & Business Media. 2.5.3

[Yun-tao et al., 2005] Yun-tao, Z., Ling, G., and Yong-cheng, W. (2005). An improved tf-idf approach for text classification. *Journal of Zhejiang University-Science A*, 6(1):49–55. 2.4.2

[Zahara, 2021] Zahara (2021). *PUTA*. G.O.Z.Z Records (2021). https://open.spotify.com/album/2ctwMRxRCZabremU4pGcz4?si=ezYCe71EQHKsVk3d6zfXew&dl_branch=1. 5

[Řehůřek, 2020] Řehůřek, R. (2020). Doc2vec. https://radimrehurek.com/gensim/models/doc2vec.html. 3.2

Anexo

Configuración de los modelos

En este Anexo se van a detallar algunos los valores de los hiperparámteros más representativos que definen los modelos utilizados. Estos valores se corresponden con los hiperparámetros que se han empleado para llegar a los resultados que se muestran a lo largo del capítulo 4.

En la Tabla 10 se muestran los valores de los hiperparámetros con los que se obtienen los resultados para la Task A (vistos en la sección 4.1). Si se aplican los modelos con dichos hiperparámetros se llega a los resultados que se ven en la Tabla 3.

En las Tablas 11 y 12 se muestran los hiperparámetros correspondientes a la Task B (sección 4.2). En concreto, en la Tabla 11 se muestran los hiperparámetros empleados para la tarea de clasificación según objetivo (4.2.1). Con estos valores se obtienen los valores vistos en las Tablas 4 y 5. En la Tabla 12 se muestran los hiperparámetros empleados para la tarea de clasificación según el tema (4.2.2). Con estos valores se obtienen los valores vistos en las Tablas 6 y 7.

Para el conjunto de redes neuronales, se ha utilizado redes de tipo *Secuencial* con capas de tipo *Dense*. La capa de entrada se compone de ocho neuronas mientras que la salida se compone de una para los estudios de la Task A y la clasificación según objetivo de la Task B y de cinco para la clasificación según tema de la Task B.

Se ha tomado como función de activación *ReLU* en la capa de entrada y en el conjunto de capas ocultas. En el caso de las clasificaciones binarias (como son las correspondientes a las Task A y a la clasificación según objetivo de la Task B), se ha empleado como función de activación para la capa de salida la función *Sigmoide*. En el caso de la clasificación multiclase como es la clasificación según tema de la Task B, se ha emplea-

do como función de activación para la capa de salida la función *Softmax*. La función de pérdida que se ha empleado en todos los casos para la construcción de la red neuronal es la función *entropía cruzada*.

En en el apartado de red neuronal de las Tablas 10, 11 y 12 se detallan los valores del número de capas ocultas (*n_hidden*) y el número de neuronas de cada capa oculta (*n_neurons*) con las que se obtiene la mejor configuración y la tasa de aprendizaje (*learning_rate*).

Para los modelos de combinación de modelo, se utiliza como entrada los mejores resultados encontrados para los modelos anteriores (*Naive Bayes*, *Regresión Logística*, *Support Vector Machine*, *Random Forest* y *Red Neuronal*). Se ha utilizado el tipo de votación *soft voting* junto con una versión de voto basado en la mayoría de voto. Ambos conceptos están explicados en la sección 2.5.6.

Tabla 10: Tabla resumen en la que se muestran los valores de hiperparámetros empleados en los modelos de clasificación para la Task A. Con ellos se obtienen los resultados que se muestran en la Tabla 3

		Castellano	Inglés
TD-IDF	Naive Bayes	var_smoothing=0.023	var_smoothing=3.265
	Regresión Logística	C=1.0, max_iter=1500, solver=newtoncg	C=0.8368, max_iter=, solver=liblinear
	SVM	C=0.889, gamma=0.6, kernel=linear	C=0.747, gamma=1e-08, kernel=linear
	Random Forest	max_depth=5305, max_features=sqrt	max_depth=1643, max_features=auto
		n_estimators=46	n_estimators=10
	Red neuronal	learning_rate=0.1, n_hidden=2, n_neurons=200	learning_rate=0.05, n_hidden=4, n_neurons=100
Bag of Words	Naive Bayes	var_smoothing=0.00068	var_smoothing=0.5267
	Regresión Logística	C=1.0, max_iter=1500, solver=newtoncg	C=0.359, max_iter=1500, solver=liblinear
	SVM	C=0.5, gamma=0.5, kernel=rfb	C=0.138, gamma=1e-05, kernel=linear
	Random Forest	max_depth=4629, max_features=auto	max_depth=60, max_features=auto,
		n_estimators=600	n_estimators=2000
	Red neuronal	learning_rate=0.05, n_hidden=3, n_neurons=200	learning_rate=0.06, n_hidden=4, n_neurons=100
Doc2Vec	Naive Bayes	var_smoothing=0.0621	var_smoothing=0.408
	Regresión Logística	C=8.858, max_iter=1500, solver=newton-cg	C=12.915, max_iter=1500, solver=newton-cg
	SVM	C=9.444, gamma=0, kernel=linear	C=7.333, gamma=1e-10, kernel=linear
	Random Forest	max_depth=6260, max_features=auto	max_depth=25, max_features=auto
		n_estimators=600	n_estimators=3500
	Red neuronal	learning_rate=0.05, n_hidden=3, n_neurons=200	learning_rate=0.03, n_hidden=2, n_neurons=200

Tabla 11: Tabla resumen en la que se muestran los valores de hiperparámetros empleados en los modelos de clasificación según objetivo para la Task B.

	Castellano	Inglés
Naive Bayes	var_smoothing=0.625	var_smoothing=0.625
Regresión Logística	C=12.915, solver=newton-cg	C=1, solver=newton-cg
SVM	C=1000, gamma=0.25, kernel=rbf	C=2.632, gamma=1.578, kernel=rbf
Random Forest	max_depth=1048576, n_estimators=500	max_depth=32, n_estimators=200
Red neuronal	learning_rate=0.005, n_hidden=2, n_neurons=100	learning_rate=0.005, n_hidden=2, n_neurons=100

Tabla 12: Tabla resumen en la que se muestran los valores de hiperparámetros empleados en los modelos de clasificación según tema para la Task B.

	Castellano	Inglés
Naive Bayes	var_smoothing=0.369	var_smoothing=0.527
Regresión Logística	C=7.592, solver= newton-cg	C=3.755, solver=newton-cg
SVM	C=1.4285, gamma=auto, kernel=linear	C=0.816, gamma=auto, kernel=linear
Random Forest	max_depth=106528681.3, n_estimators=571	max_depth=1941700, n_estimators=533
Red neuronal	learning_rate=0.01, n_hidden=5, n_neurons=200	learning_rate=0.003, n_hidden=2, n_neurons=50