UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

Máster
Ingeniería y Ciencia de Datos

TFM: CLASIFICACIÓN DE IDEOLOGÍA POLÍTICA EN TEXTOS

Autor: JUAN CARLOS PÉREZ GONZÁLEZ

Director: ALVARO RODRIGO YUSTE

Codirector: ROBERTO CENTENO SANCHEZ

09/2022

Resumen

El presente trabajo nos permite explorar las posibilidades de clasificación de textos de forma automática, teniendo como objetivo principal detectar la ideología política de los autores a partir de sus textos (obtenidos de la red social Twitter). Partimos de los datos publicados en la competición "IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology. La colección de la tarea recopila tweets publicados ....fueron recopilados durante 2020 y 2021 de las cuentas de Twitter de políticos y periodistas políticos en donde las menciones a cuentas de Twitter y a partidos políticos han sido anonimizadas.

La tarea se aborda desde una perspectiva de procesamiento del lenguaje natural, donde llevamos a cabo una exploración inicial de los textos de entrada (analizando métricas, palabras empleadas, …), una normalización de los textos (eliminación de stopwords, lemmatizacion, ...) y a partir de los datos normalizados se entrenan y evalúan distintos modelos de clasificación (regresión logística, redes neuronales, SVM, Random Forest, LGBM Classifier, BERT, MLPClassifier).

Los mejores resultados se consiguen empleando redes neuronales para la clasificación de la ideología política tanto binaria como multiclase y LGBM Classifier para la profesión y el género y por lo general, obtenemos un f1-score promedio de 0.869236, bastante por encima de los alcanzados por el método baseline de la tarea original.

Palabras clave

Procesamiento del lenguaje natural, clasificación de textos, algoritmos de clasificación, identificación ideología política.

Abstract

This research allows us explore the possibilities of classifying texts automatically. The main target is detecting the political ideology of the authors from their texts (tweets from the social network Twitter), based on the data from challege “IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology”. The dataset was collected during 2020 and 2021 from the Twitter accounts of politicians and political journalists in Spain, where the Twitter accounts of the politicians and political parties were anonymised.

The task is approached from a natural language processing perspective. We carry out an initial exploration of the input texts (analyzing metrics, words,...), we normalize the texts (stopwords, lemmatization,...) and we train and evaluate different classification models using the normalized data (logistic regression, neural networks, SVM, Random Forest, LGBM Classifier, BERT, MLPClassifier).

The best results are achieved using neural networks for the classification of political ideology, both binary and multiclass, and LGBM Classifier for profession and gender. We obtain results well, average f1-score of 0.869236, above those achieved by the baseline method of the original task.

Keywords

Natural Language Processing, text classification, classification algorithms, political ideology identification.

Índice

Índice de figuras

Índice de tablas

1. INTRODUCCIÓN Y OBJETIVOS

La ideología política es un rasgo psicográfico que se puede utilizar para comprender el comportamiento individual y social, incluidos los valores morales y éticos, así como las actitudes, valores, sesgos y prejuicios inherentes (Verhulst et al., 2012). La relación entre los rasgos de personalidad y la ideología política fue demostrada en Fatke (2017), que recopiló datos de 21 países y encontró una correlación entre la ideología política y los cinco grandes rasgos de personalidad.

En este trabajo evaluaremos diferentes métodos para clasificar la ideología política como rasgo psicográfico y el género y la profesión como rasgos demográficos a partir de un conjunto de tweets publicados por usuario. Los métodos propuestos se evaluarán usando las colecciones de la competición "IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology" así podremos comparar nuestros resultados con los obtenidos por distintos investigadores sobre una colección actual.

Abordaremos el trabajo desde una perspectiva de procesamiento del lenguaje natural donde descompondremos el problema en cuatro problemas de clasificación (género, profesión, e ideología política binaria y multiclase), para lo cual llevaremos a cabo una exploración inicial de los textos de entrada, una normalización de los textos y finalmente entrenaremos y evaluaremos distintos modelos de clasificación.

1.1 Motivación

La motivación principal para realizar este trabajo fin de Master (TFM) subyace del hecho de poder extraer un rasgo psicográfico, como es la ideología política, del autor de un conjunto de textos. Estos rasgos psicográficos están relacionados con los rasgos de personalidad y se pueden utilizar para comprender el comportamiento individual y social, incluidos los valores morales y éticos, así como las actitudes, valoraciones, sesgos y prejuicios inherentes. Podemos aplicar los resultados a profundizar en el estudio del comportamiento humano y en una variedad de aplicaciones como la adaptación de servicios de atención al cliente o campañas publicitarias a medida.

Otra de las motivaciones proviene del hecho de que en la actualidad se dispone de una cantidad considerable de información presente en textos (procedentes de redes sociales (Twitter, Facebook, …), textos legales, contratos, partes de siniestros, …), información presentada de forma desestructurada que requiere de un esfuerzo manual considerable para poder clasificarla, analizarla y extraer información de ellos.

El presente TFM no solo nos permite explorar las posibilidades de clasificación de textos de forma automática sino también nos permite ir un paso más allá al extraer la ideología política como rasgo psicográfico lo que le da un potencial mucho mayor. Con ello podemos poner en práctica no solo los conocimientos adquiridos durante la realización del Máster de Ingeniería y Ciencia de Datos, sino también podremos extrapolar las lecciones aprendidas durante la realización del presente TFM al mundo laboral.

1.2 Objetivos

El objetivo de este trabajo es evaluar y comparar el uso de distintos métodos para detectar la ideología política, el género y la profesión de un usuario a partir de sus Tweets. Para evaluar nuestros métodos usaremos la colección de la competición "IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology". Nos centraremos no solo en la detección, sino que evaluaremos diferentes modelos de clasificación para analizar sus resultados, siendo este su objetivo principal. Para ello dividiremos nuestro objetivo en los siguientes objetivos secundarios:

- Analizar las técnicas actuales utilizadas para la clasificación automática de textos.
- Analizar los datos de entrada disponibles en el dataset de la tarea.
- Normalizar los datos de entrada
- Evaluar diferentes algoritmos con distintas configuraciones
- Seleccionar el algoritmo que presente mejores resultados para cada clasificación.

1.3 Propuesta

Expondremos en los siguientes apartados el trabajo realizado para cumplir con los objetivos mencionados anteriormente:

- Apartado 2. Estado del arte: describiremos que soluciones se están usando en problemas similares.
- Apartado 3. Exploración y análisis de los datos: partimos del conjunto de datos suministrados por la competición "IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology". Analizaremos dichos datasets para ver cómo están distribuidas las clases (según su género, profesión e ideología política), que palabras utilizan más cada clase, sacaremos métricas de los textos de los tweets con los que podremos obtener que clase utiliza más verbos, cual usa más adjetivos, la que emplea frases más complejas, …
- Apartado 4. Normalización de los tweets de entrada: explicaremos las transformaciones que hemos aplicado al texto original de los tweets, con el objetivo de facilitar posteriores procesos. Con esta transformación pasaremos los textos a minúsculas, eliminaremos stopwords y emojis y estudiaremos si realizamos un proceso de stemming o lematización.
- Apartado 5. Modelos de predicción y clasificación: detallaremos los distintos algoritmos de clasificación empleados, donde hemos abarcado redes neuronales, arboles de decisión, Transformers, regresión logística, …
- Apartado 6 . Evaluación de los resultados: evaluaremos los resultados obtenidos por los diferentes algoritmos de clasificación, utilizando diferentes métricas (accuracy, f1-score macro, …). Debemos tener en cuenta que la métrica

empleada para clasificar los trabajos en la tarea va a ser la f1-score por lo que le daremos más importancia. Los distintos trabajos se clasificarán utilizando la media aritmética de la f1-score obtenida para cada una de las clasificaciones (ideología política binaría y multiclase, género y profesión).

- Apartado 7. Conclusiones y trabajos futuros: donde se describen las conclusiones y una serie de tareas para mejorar nuestro trabajo.

2. ESTADO DEL ARTE

En este apartado, se realiza un análisis del contexto en el que se enmarca el presente trabajo. Para ello, revisaremos trabajos similares al propuesto y a continuación nos centraremos en los trabajos presentados a la competición "IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology" y que mejor puntuación han obtenido en la fase de evaluación.

2.1 Trabajos similares

No se han encontrado muchos trabajos similares relacionados con la extracción de la ideología política en otros idiomas, si bien si existen algunos ejemplos:

"Automatic categorization of Arabic articles based on their political orientation" [11] donde se extrae orientación política a partir de un corpus de artículos en árabe que fueron recolectados y etiquetados manualmente. Los mejores resultados de dicho trabajo se obtienen empleando SVM.

"Automatic prediction of gender, political affiliation, and age in Swedish politicians from the wording of their speeches—A comparative study of classifiability" [12] que explora la clasificación automática de los políticos suecos a partir de sus discursos (realizados entre 2003 y 2010 en el parlamento) en clases, según los rasgos personales (género, edad y afiliación política). Mediante el uso de SVM se logra alcanzar una precisión del 81,2% para el género, 89,4% para la afiliación política binaria y 78,9% para la edad.

"Computational methods in authorship attribution" [13] en el cual se examina la afiliación ideológica y organizacional como dos problemas independientes. Se emplean dos corpus (uno de 552 documentos y otro de 1485 etiquetados con cuatro ideologías políticas) relacionados con documentos religiosos y políticos en árabe.

"We can detect your bias: Predicting the political ideology of news articles" [14] donde se extrae la ideología política principal a partir de artículos periodísticos. Para ello emplearon un conjunto de datos de 34.737 artículos anotados manualmente según la ideología política (izquierda, centro o derecha).

Como se ve existen trabajos previos relacionados realizados en diferentes idiomas, pero en español no encontramos ningún caso similar. También habría que señalar que en dos de los trabajos se indica que han empleado SVM, por lo que sería un buen candidato para emplear en la fase de evaluación.

2.2 Trabajos presentados a la competición IberLEF 2022 Task PoliticEs

En este apartado describiremos los trabajos presentados que mejor puntuación han obtenido en la fase de evaluación de la competición.

Political Author Profiling using BETO and MarIA [15]
Propone un sistema combinando dos modelos basados en transformadores preentrenados en castellano, BETO (modelo en español de BERT) y MarIA (modelo basado en RoBERTa presentado recientemente que ha sido preentrenado usando un corpus masivo de 570 GB de textos limpios con 135 mil millones de palabras extraídas

del Archivo Web Español rastreado por la Biblioteca Nacional de España entre 2009 y 2019), con el que se obtiene el mejor valor promedio de f1-score. La arquitectura propuesta en el trabajo se puede ver en la Ilustración 1:

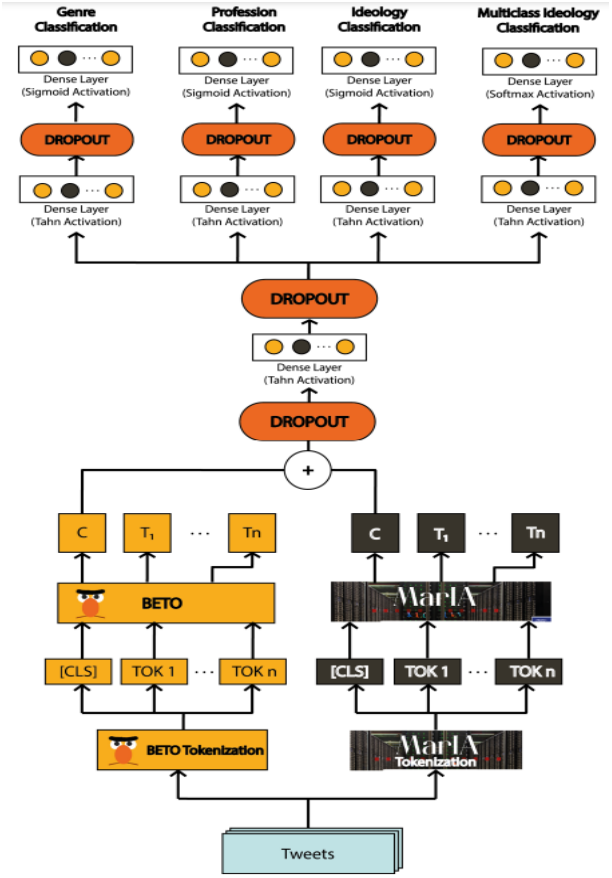

Ilustración 1. Arquitectura propuesta[15]

Los tweets de los usuarios se transforman en tokens que sirven como entrada a los modelos BETO y MarIA. Cada modelo genera un vector de 768 dimensiones que representa el significado de toda la oración. Ambos vectores se concatenan y el resultado sirve como entrada a cuatro bloques de clasificación separados, uno para cada tarea de clasificación. Cada bloque de clasificación está formado por una capa densa completamente conectada con 768 unidades y función de activación $tanh$ que conecta directamente con la capa superior softmax o sigmoide, que devuelve una puntuación de probabilidad para cada clase. Las capas dropout se emplean para evitar sobre ajuste.

Debido a las limitaciones de los modelos basados en BERT, donde se establece el número máximo de tokens a 512, los tweets de los mismos usuarios se agrupan en bloques de 512 tokens. Cada bloque se clasifica individualmente utilizando la arquitectura descrita anteriormente y se emplea un sistema de votación para obtener la clasificación final.

PolitiBETO, a Domain- Adapted Transformer for Multi-class Political [16] es uno de los seleccionados para ser presentados en el IberLEF 2022 obtiene la segunda mejor puntuación promedia 0.890961. En este trabajo se propone el modelo "PolitiBETO" que se construye mediante una Adaptación de Dominio en dos etapas sobre BETO con la

idea de enseñarle a BETO la estructura del lenguaje en Twitter y especializarlo aún más para un lenguaje político. Ambas Adaptaciones de Dominio se realizan siguiendo el procedimiento sugerido para AdaptaBERT que consiste en continuar el preentrenamiento de BETO a través de la tarea de Modelado de Lenguaje Enmascarado para más epochs.

Towards Robust Spanish Author Profiling and Lessons Learned from Adversarial Attacks [17] obtiene la tercera mejor puntuación con un f1-score promedio de 0.889182. Utiliza regresión logística regularizada L2 para la clasificación de todas las clases empleando una serie de características léxicas y estilísticas calculadas sobre el conjunto de tweets de cada autor:

- N-gramas de palabras: 1-4 frecuencias de n-gramas extraídas en minúsculas
- N-gramas de caracteres: TF-IDF sobre unigramas de caracteres en minúsculas;
- Indices de Legibilidad: Kincaid, ARI, Coleman-Liau, facilidad de lectura de Flesch, Gunning-Fog, LIX, Índices SMOG, RIX y Dale-Chall para caracterizar el estilo de escritura y complejidad textual

Ensemble Based Classification Algorithms for Author Profiling in Spanish Language [18],logra alcanzar el cuarto puesto de la competición con un f1-score promedio de 0.879757. Se emplea BETO para el género y la profesión, XGBoost para ideología binaria y Regresión logística para ideología multiclase. Para evitar la limitación de 512 tokens de BERT agrupan el conjunto de tweets de cada autor en grupos de 12 tweets para el género y 9 para la profesión.

Como vemos todos los trabajos estudiados excepto uno emplea BERT en español y para evitar el problema de la limitación de tamaño en BERT dividen en bloques los tweets de cada usuario. Además, casi todos los trabajos indican explícitamente que se ha realizado un trabajo previo de preprocesamiento similar al que realizamos nosotros para normalizar los tweets (pasar a minúsculas, eliminar signos de puntuación, reemplazo/eliminación de emojis,…). También vemos que en solo uno de los trabajos se emplean las métricas de los textos, índices de legibilidad, y en solo otro de ellos se emplea un clasificador diferente para cada clasificación. En nuestro trabajo además de realizar la clasificación completa, analizaremos la colección suministrada y evaluaremos distintos métodos para la clasificación de cada tipo.

3. EXPLORACION Y ANALISIS DATOS

Para el presente trabajo partimos del conjunto de datos suministrados en la competición "IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology" los cuales fueron recopilados durante 2020 y 2021 de las cuentas de Twitter de políticos y periodistas políticos utilizando el UMUCorpusClassifier(García-Díaz et al., 2020). Estos datasets presentan los siguientes datos:

- label: indica el usuario anonimizado de Twitter de cada autor
- gender: género masculino o femenino del autor
- profession: la profesión del autor, político o periodista
- ideology_binary: indica si el autor es de derechas o izquierdas
- ideology_multiclass: clasificación multiclase de la ideología política, izquierdas, izquierda moderada, derechas o derecha moderada
- tweet: texto del tweet del autor

Las cuentas de los políticos fueron seleccionadas entre: miembros del gobierno de España, miembros del Congreso y Senado de España, alcaldes de algunas ciudades importantes de España, presidentes de las comunidades autónomas, expolíticos, y colaboradores afiliados a partidos políticos. Mientras que se seleccionaron periodistas de diferentes medios españoles, como ABC, El País, El Mundo, La Razón, ... Para la construcción del dataset se descartaron los tweets que comparten contenido de sitios web de noticias sin usar retweets y aquellos que contienen menciones a sitios de noticias o algunas pistas lingüísticas, como el símbolo de la pipa, que los sitios de noticias usan comúnmente para categorizar sus noticias. Las cuentas de Twitter fueron anonimizadas reemplazándolas con el token @user y los nombres de partidos políticos por [political_party]. En consecuencia, los rasgos del autor no se pueden adivinar trivialmente leyendo su nombre y buscando información sobre ellos en Internet.

Así el dataset de entrenamiento consta de 37.560 tweets de 313 usuarios distintos con una fila por tweet (todos los usuarios tienen 120 tweets distintos, es decir hay 120 filas de cada usuario), por lo que agruparemos los tweets por autor y realizaremos una clasificación a nivel de autor tal y como se indica en la competición y además realizaremos una clasificación extra a nivel de tweet. A continuación, se muestra una tabla resumen del número de ejemplos de cada tipo a clasificar que tenemos en el dataset de entrada tanto a nivel de autor ("*tabla 1*") como a nivel de tweet ("*tabla 2*"), donde se observa claramente que las clases no están balanceadas:

Ideología política		género		profesión		Ideología política multiclase			
izda	drcha	masculino	femenino	políticos	periodistas	izda	izda moderada	drcha moderada	drcha
178	135	136	177	251	62	76	101	95	41
56,87%	43,13%	43,45%	56,55%	80,19%	19,81%	24,28%	32,27%	30,35%	13,10%

Tabla 1. nº y % de autores de cada clase

Ideología política		género		profesión		Ideología política multiclase			
izda	drcha	masculino	femenino	políticos	periodistas	izda	izda moderada	drcha moderada	drcha
21.360	16.200	16.320	21.240	30.120	7.440	9.120	12.120	11.400	4.920
56,87%	43,13%	43,45%	56,55%	80,19%	19,81%	24,28%	32,27%	30,35%	13,10%

Tabla 2. nº y % de tweets de cada clase

Para ayudarnos a analizar los tweets de entrada se han empleado dos librerías que calculan una serie de métricas a partir de textos, MultiAzterTest y TextComplexityFreeling.

TextComplexityFreeling consiste en una biblioteca Python para calcular diferentes métricas de complejidad a partir de textos planos disponible en varios idiomas (inglés, español, francés, …). Con esta librería podemos calcular las siguientes métricas:

- ***Complejidad léxica***: La complejidad léxica de un texto, determinada por la frecuencia de uso y la densidad léxica, se basa en el número de palabras de contenido diferente por oración (Índice de Complejidad Léxica, LC) y en la medición del número de palabras de baja frecuencia por cada 100 palabras de contenido (Índice de Palabras de Baja Frecuencia, ILFW). En consecuencia, cuanto mayor sea el índice LC, mayor será la dificultad en la comprensión lectora.
- ***Legibilidad de Spaulding***: Comúnmente conocido como Índice SSR, fue propuesto por Spaulding en 1956 y se enfoca en medir el vocabulario y la estructura de las oraciones para predecir la dificultad relativa de la legibilidad de un texto.
- ***Complejidad de oraciones***: el índice de complejidad de oraciones (SCI) fue propuesto por Anula en 2018 como una medida de la complejidad de las oraciones en un texto literario dirigido a estudiantes de un segundo idioma. Esta medida de complejidad sintáctica se enfoca en medir el número de palabras por oración, obteniendo así el índice de longitud de oraciones (Average Sentence Length, ASL), y el número de oraciones complejas por oración, a partir de un índice de oraciones complejas (Complex Sentences, CS).
- ***Índice de Legibilidad Automatizado*** (ARI): propuesto por Senter y Smith, en 1967, es uno de los índices más utilizados debido a su facilidad de cálculo. Este índice mide la dificultad de un texto a partir del número medio de caracteres (letras y números) por palabra y el número medio de palabras por frase.
- ***Profundidad del árbol de dependencia***: esta medida fue propuesta por Saggion et al. en 2015. Es una métrica útil para capturar la complejidad sintáctica: las oraciones largas pueden ser sintácticamente complejas o contener una gran cantidad de modificadores (adjetivos, adverbios o frases adverbiales). Complementa la medida de ASL, ya que captura la complejidad sintáctica en términos de estructuras recursivas o anidadas.
- ***Signos de puntuación***: esta medida también fue propuesta por Saggion et al, el número medio de signos de puntuación se utiliza como uno de los indicadores de la sencillez del texto.
- ***Legibilidad de Fernández-Huerta***: Blanco (2002) y Ramírez (2013) proponen esta medida de complejidad como una adaptación al español de la prueba de legibilidad de Flesch (Flesch, 1948).

- ***Legibilidad de Flesch-Szigrist*** (IFSZ): Los trabajos de Barrio et al. (2008) y Ramírez et al. (2013) proponen el índice de legibilidad de Flesch-Szigristzt como una modificación de la fórmula de Flesch adaptada al español por Szigriszt-Pazos en 1993. Este índice es considerado actualmente un referente para el idioma español. Se enfoca en medir el número de sílabas por palabra y el número de oraciones por palabra en el texto.
- ***Comprensibilidad de Gutiérrez de Polini***: Esta métrica, desarrollada originalmente en 1972 fue creada desde un principio para el español (Rodríguez, 1980). Se enfoca en medir el promedio de letras por palabra y el promedio de palabras por oración.
- ***Legibilidad Mu***: Es una fórmula para calcular la legibilidad de un texto, desarrollado por Muñoz en 2006. Esta medida se centra en medir el número de palabras, el número medio de letras por palabra y su varianza.
- ***Edad mínima para comprender***: Es una adaptación al español de la fórmula original de Flesch para el inglés. Mide el número medio de sílabas por palabra y el número medio de palabras por frase para obtener la edad mínima necesaria para comprender un texto.
- ***Legibilidad SOL***: Contreras et al. (1999) propone la métrica SOL como una adaptación al español de la fórmula SMOG propuesta por Mc Laughlin (1969). Mide la legibilidad de un texto por medio del nivel de grado, que es el número de años de escolaridad necesarios para comprender el texto.
- ***Años Crawford***: Esta medida fue propuesta por Alan N. Crawford en 1989. Se utiliza para calcular los años de escolaridad necesarios para comprender un texto. Mide el número de frases por cada cien palabras y el número de sílabas por cada cien palabras.

MultiAzterTest es una herramienta de PNL de código abierto que analiza textos en más de 125 medidas de cohesión, idioma y legibilidad. Está disponible para los idiomas inglés, español y euskera. Con ella podemos calcular características lingüísticas y estilísticas, donde las características lingüísticas son las relacionadas con la morfología, la sintaxis y la semántica mientras que los rasgos estilísticos están relacionados con la cohesión, el conocimiento del vocabulario, … Con ella podemos obtener:

- ***Funciones descriptivas***: MultiAzterTest proporciona índices descriptivos para ayudar al usuario a verificar la salida e interpretar patrones de datos (número y proporciones de letras, sílabas, lemas, palabras, oraciones y párrafos, ...)
- ***Diversidad léxica***: Estas métricas analizan las diferentes palabras utilizadas en el texto (densidad léxica, densidades de sustantivos, verbos, adjetivos y adverbios; Densidad léxica Honoré, Densidad léxica Maas, …)
- ***Fórmulas clásicas de legibilidad***: calcula algunas de las fórmulas más comunes como legibilidad de Flesch para inglés, Legibilidad de Fernández-Huerta para español y el grado SMOG para inglés.
- ***Frecuencias de palabras***: MultiAzterTest obtiene las frecuencias de palabras y calcula una lista de métricas (mínima frecuencia de palabras por oración; número e incidencia de palabras raras, …)
- ***Conocimiento de vocabulario***: calcula el número y la incidencia de palabras en cada Nivel MCER (Common European Framework of Reference for Languages), solo disponible en inglés

- **Información morfológica de las palabras**: Estas métricas analizan la forma y estructura de las palabras.
- ***Sintaxis***: estas métricas se centran en el gobierno y la estructura de las oraciones (sintagmas nominales y sintagmas verbales por oración; número e incidencia de las oraciones subordinadas, pasivas y de negación, ...)
- ***Información semántica***: se tienen en cuenta los valores medios de la polisemia y los valores medios de los valores hiperónimos.
- **Superposición semántica** (similitud semántica) entre oraciones o entre párrafos
- ***Cohesión referencial***
- ***Cohesión lógica***

3.1 Analisis Dataset Entrada

Como ya se indicó, el dataset de entrada presenta 313 autores con 120 tweets por autor, es decir tenemos un total de 37.560 tweets diferentes. En la *tabla 3* observamos que los datos no están balanceados para ninguna clasificación, tenemos más autores de izquierdas (un 56,87% de los casos), más de género masculino (el 56,55%), muchos más políticos (80,35%) que periodistas y más ejemplos de ideología moderada de izquierdas (32,27%) que del resto de sus categorías:

	clase	nº tweets	%
Ideología binaria	left	21.360	56,87%
	right	16.200	43,13%
género	female	16.320	43,45%
	male	21.240	56,55%
profesión	journalist	7.381	19,65%
	politician	30.179	80,35%
Ideología multiclase	left	9.120	24,28%
	moderate_left	12.120	32,27%
	moderate_right	11.400	30,35%
	right	4.920	13,10%

Tabla 3. Nº ejemplos y porcentaje de cada tipo de clasificación

El siguiente paso fue realizar un primer análisis de los textos de entrada según la clase a la que pertenecen, obteniendo las palabras más usadas, cloudwords y métricas de los textos empleando las librerías anteriormente descritas, MultiAzterTest y TextComplexityFreeling.

3.1.1 Ideología Binaria

En este apartado vamos a realizar el análisis de los textos de los autores según su orientación política binaria, izquierdas o de derechas. Comenzamos mostrando la lista de palabras más usadas *tabla 4* con la lista de los autores de izquierdas y *tabla 5* con las de derechas), así como la nube de palabras de cada ideología binaría.

Izquierdas (left)

Top 10 palabras más usadas (izdas):

	palabra	Nº apariciones
1	gobierno	2665
2	españa	2031
3	años	1835
4	gracias	1322
5	país	1300
6	madrid	1189
7	personas	1078
8	vida	936
9	política	929
10	gente	910

Tabla 4. Palabras más usadas en los tweets de autores de izquierdas

Derechas (right)

Top 10 palabras más usadas (drchas):

	palabra	Nº apariciones
1	gobierno	2948
2	españa	2382
3	sánchez	1689
4	años	1239
5	españoles	1021
6	gracias	960
7	madrid	843
8	libertad	683
9	ley	614
10	presidente	600

Tabla 5. Palabras más usadas en los tweets de autores de derechas

CloudWords(izdas)

Ilustración 2. Nube de palabras de los tweets de autores de izquierdas

CloudWords (drchas)

Ilustración 3. Nube de palabras de los tweets de autores de derechas

Se observan diferencias entre la lista de más usadas y la nube de palabras según sea la ideología de los usuarios de derechas o de izquierdas. Si bien en ambos casos las palabras más empleadas son “gobierno” y “España”, en la lista de los usuarios de izquierdas aparecen palabras (“país”, “personas”, “gente”, “política” y “vida”) que no aparecen en la lista de las más empleadas por los de derechas. Mientras que en el top de los de derechas aparecen algunas que no están en la lista de izquierdas (“Sanchez”, ”presidente”, ”españoles”, “libertad” y “ley”), estando “sanchez” como la tercera más usada y apareciendo “presidente” en el puesto 10, siendo una manera de referirse a él, al presidente del gobierno Pedro Sanchez. En el “CloudWords” de la derecha encontramos palabras como “ETA”, “Cataluña”, “Izquierda”, … mientras que en el de la izquierda aparecen “mujeres”, “medida”, “lucha”, ….

A continuación, podremos observar en la tabla 6 y en la tabla 7 un resumen con la media y la desviación estándar de las métricas de los textos de autores de izquierdas y de derechas respectivamente. En las ilustraciones 4 y 5 mostramos la distribución de valores de las métricas de los tweets.

Media y desviación estándar de las métricas calculadas más representativas (izquierdas)

Métrica	mean	std
LC	4,6212	2,4101
SSR	156,5716	32,7857
SCI	16,6988	10,9603
ARI	12,2868	6,5620
MeanEmdedding	6,8850	1,4338
Huerta	73,9550	19,4434
IFSZ	57,5991	21,5267
Polini	41,2869	9,0231
Mu	51,0515	65,5720
MinAge	11,1078	3,2408
SOL	9,6185	2,9441
Crawford	5,3978	1,7064
Num_adj	2,4213	1,7354
num_adv	1,8087	1,6221
num_noun	8,1548	3,2335
num_verb	5,1029	2,6386
num_dif_rare_words	3,9537	2,1888

Tabla 6. Métricas de los tweets de autores de izquierdas

*en las celdas con una diferencia de media mayor a un punto

Donde:

- LC → Complejidad Lexica
- SSR → Legibilidad de Spaulding
- SCI → Índice de complejidad de oraciones
- ARI → Índice de Legibilidad Automatizado
- MeanEmdedding →Profundidad del árbol de dependencia
- Huerta → Legibilidad de Fernández-Huerta
- IFSZ → Legibilidad de Flesch-Szigrist
- Polini → Comprensibilidad de Gutiérrez de Polini

Media y desviación estándar de las métricas calculadas más representativas (derechas)

Métrica	mean	std
LC	4,5914	2,3648
SSR	157,6947	31,9713
SCI	16,6112	10,7834
ARI	12,0409	6,1942
MeanEmdedding	6,8788	1,3928
Huerta	76,1150	21,1702
IFSZ	59,8452	22,2220
Polini	41,7259	8,0408
Mu	49,2115	58,0861
MinAge	10,8681	3,2523
SOL	9,5023	2,8603
Crawford	5,2321	1,6640
num_adj	2,3151	1,6540
num_adv	1,8375	1,5785
num_noun	8,1628	3,0840
num_verb	5,4034	2,5781
num_dif_rare_words	4,2005	2,2576

Tabla 7. Métricas de los tweets de autores de izquierdas

- Mu → Legibilidad Mu
- MinAge → Edad mínima para comprender
- SOL → Legibilidad SOL
- Crawford → Años Crawford
- num_adj → nº adjetivos
- num_adv → nº adverbios
- num_noun → nº nombres
- num_verb → nº verbos
- num_dif_rare_words → nº palabras raras

Gráfica de las métricas (izquierdas)

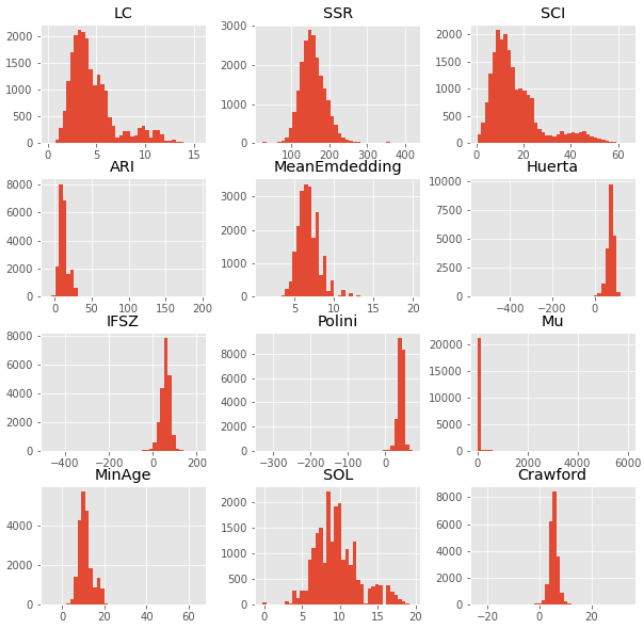

Ilustración 4. Gráficas de las métricas de los tweets de autores de izquierdas

Gráfica de las métricas (derechas)

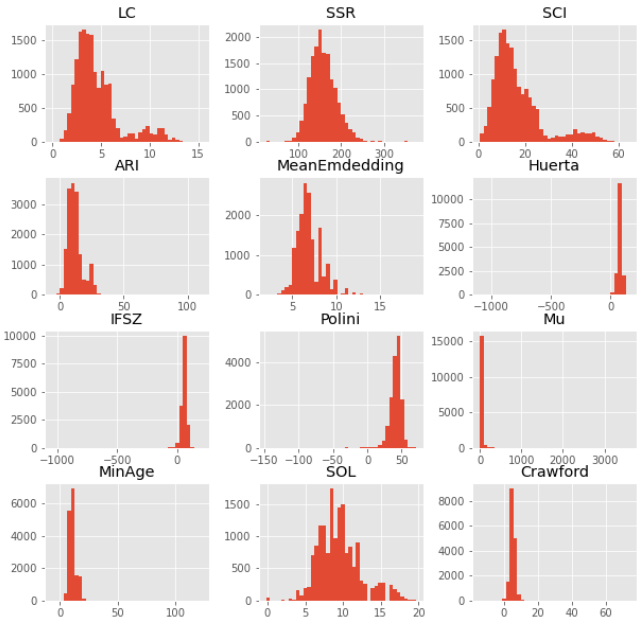

Ilustración 5. Gráficas de las métricas de los tweets de autores de derechas

Se puede ver que los valores de las métricas de los textos son bastantes similares, salvo los índices SSR, Huerta (Legibilidad de Fernández-Huerta) y IFSZ (donde para los usuarios de derechas es entre dos y tres puntos superiores) y legibilidad Mu que es casi dos puntos superiores en los de izquierdas. Los usuarios de derechas emplean de media un número ligeramente mayor de adverbios, nombres, verbos y "palabras raras" mientras que los de izquierdas presentan un número ligeramente superior de adjetivos.

3.1.2 Género

En este apartado vamos a realizar el análisis de los textos de los autores según su género, masculino o de femenino. Primeramente, mostramos la lista de palabras más usadas por cada clase (*tabla 8* con la lista del género masculino y tabla 9 con las del femenino), así como la nube de las palabras empleadas por cada una de ellas.

Masculino (male)

En la "tabla 8" mostramos las 10 palabras más usadas por el género masculino:

	palabra	Nº apariciones
1	gobierno	3168
2	españa	2743
3	años	1879
4	sánchez	1269
5	estado	1199
6	madrid	1139
7	gracias	1065
8	país	1011
9	personas	828
10	política	823

Tabla 8. Palabras más usadas en los tweets de autores masculinos

Femenino (female)

En la "tabla 9" mostramos las 10 palabras más usadas por el género masculino:

	palabra	Nº apariciones
1	gobierno	2445
2	españa	1670
3	gracias	1217
4	años	1195
5	madrid	893
6	país	885
7	estado	839
8	personas	818
9	sánchez	793
10	mujeres	771

Tabla 9. Palabras más usadas en los tweets de autoras femeninas

CloudWords (male)

Ilustración 6. Nube de palabras de los tweets de autores masculinos

CloudWords (female)

Ilustración 7. Nube de palabras de los tweets de autoras femeninas

Tanto la nube de palabras empleadas como la lista de 10 de palabras más usadas es bastante similar en ambos casos. Si bien para el género femenino aparece la palabra "mujeres" en el top 10 mientras que para el género masculino no aparece ni si quiera en la nube de palabras empleadas. En el top10 del género masculino "Sanchez" aparece en el 4º puesto mientras que en la lista de mujeres baja al 9º puesto. La única palabra que aparece entre las 10 más usadas por el género masculino que no aparece en la lista del género femenino es "política".

Seguidamente mostramos la *"tabla 10"* y la *"tabla 11"* con un resumen con la media y la desviación estándar de las métricas de los textos de autores masculinos y femeninos respectivamente, junto a las ilustraciones 8 y 9 donde mostramos la distribución de valores de las métricas de los tweets según el género.

Media y desviación estándar de las métricas calculadas más representativas (male)

Métrica	mean	std
LC	4,6640	2,4293
SSR	156,7872	31,5863
SCI	16,9636	11,0461
ARI	12,1643	6,4759
MeanEmdedding	6,9256	1,4214
Huerta	76,0968	19,7451
IFSZ	59,3220	21,8151
Polini	41,6458	8,5283
Mu	49,4853	67,1851
MinAge	10,9722	3,2846
SOL	9,6046	2,9174
Crawford	5,2779	1,6500
num_adj	2,4121	1,7067
num_adv	1,8515	1,6118
num_noun	8,2441	3,1095
num_verb	5,3608	2,5909
num_dif_rare_words	4,1297	2,2243

Tabla 10. Métricas de los tweets publicados por autoras femeninas

Media y desviación estándar de las métricas calculadas más representativas (female)

Métrica	mean	std
LC	4,5359	2,3375
SSR	157,4059	33,5191
SCI	16,2672	10,6575
ARI	12,2021	6,3164
MeanEmdedding	6,8260	1,4076
Huerta	73,3115	20,7490
IFSZ	57,5863	21,8739
Polini	41,2556	8,7237
Mu	51,2634	55,6974
MinAge	11,0463	3,1991
SOL	9,5212	2,8969
Crawford	5,3894	1,7392
num_adj	2,3279	1,6936
num_adv	1,7816	1,5917
num_noun	8,0466	3,2434
num_verb	5,0655	2,6410
num_dif_rare_words	3,9695	2,2160

Tabla 11. Métricas de los tweets publicados por autores masculinos

*en las celdas con una diferencia de media mayor a un punto con respecto al resto de clases

Gráfica de las métricas (male)

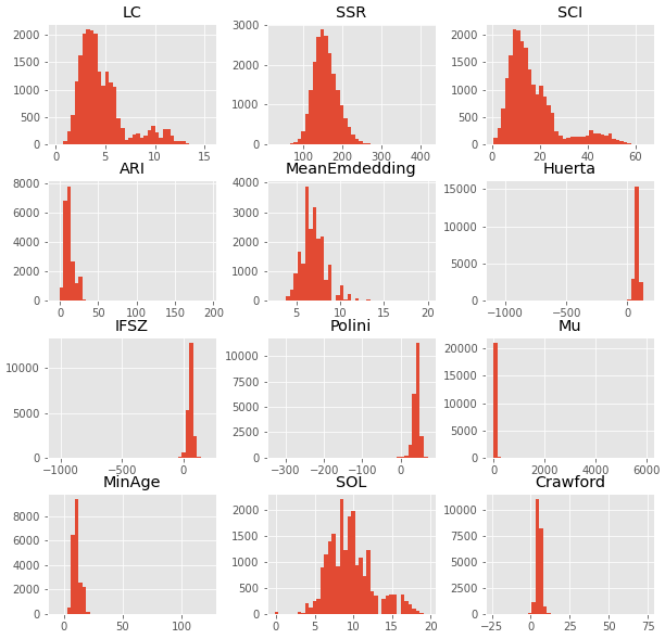

Ilustración 8. Gráficas de las métricas de los tweets de autores masculinos

Gráfica de las métricas (female)

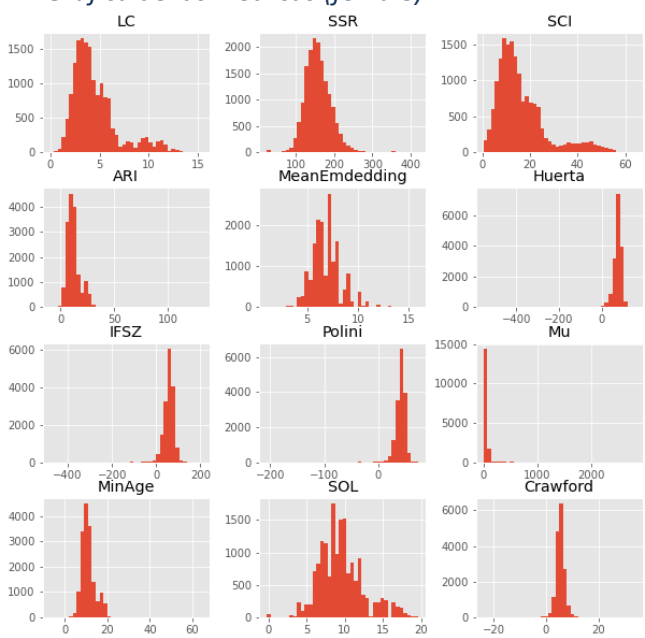

Ilustración 9. Gráficas de las métricas de los tweets de autoras femeninas

Las métricas son bastantes similares, si bien se observa que el género femenino emplea un mayor número de verbos, adjetivos, nombres, adverbios y "palabras raras", presentando a su vez un índice de legibilidad de Huerta e IFSZ entre dos y 3 puntos superior al género masculino mientras que la legibilidad Mu es casi dos puntos superiores en los hombres.

3.1.3 Profesión

En este apartado vamos a realizar el análisis de los textos de los autores según su profesión, político o periodista. Comenzamos de nuevo mostrando la lista de palabras más usadas por cada clase (*tabla 12* con la lista de los políticos y tabla 13 con las de los periodistas), así como la nube de las palabras empleadas por cada una de ellas.

Político (politician)

Top 10 palabras más usadas (político):

	palabra	Nº apariciones
1	gobierno	5063
2	españa	3884
3	años	2370
4	gracias	1929
5	estado	1792
6	sánchez	1786
7	país	1643
8	madrid	1540
9	personas	1419
10	ley	1361

Tabla 12. Palabras más usadas en los tweets de políticos

Periodista (journalist)

Top 10 palabras más usadas (periodista):

	palabra	Nº apariciones
1	años	704
2	gobierno	550
3	españa	529
4	madrid	492
5	gracias	353
6	gente	313
7	sánchez	276
8	vida	264
9	país	253
10	estado	246

Tabla 13. Palabras más usadas en los tweets de periodistas

CloudWords (político)

Ilustración 10. Nube de palabras de los tweets de políticos

CloudWords (periodista)

Ilustración 11. Nube de palabras de los tweets de periodistas

Se observa que las palabras empleadas vuelven a ser bastante similares, apareciendo en los políticos "ley" y "personas" en la lista de más usadas mientras que para los periodistas nos encontramos con "gente" y "vida". En la nube de los

periodistas nos encontramos “Ayuso”, “vacuna”, “dato” que no encontramos en la de políticos, mientras que en la de políticos aparece “apoyo”, “frente”, “familia”.

A continuación, mostramos dos tablas, *tabla 14* y *tabla 15* con un resumen de la media y la desviación estándar de las métricas de los textos de políticos y de periodistas respectivamente. En las ilustraciones 12 y 13 mostramos la distribución de valores dichas métricas.

Media y desviación estándar de las métricas calculadas más representativas(político)

Métrica	mean	std
LC	4,7263	2,4122
SSR	157,8493	32,0619
SCI	17,1540	11,0093
ARI	12,6484	6,4103
MeanEmdedding	6,9279	1,4012
Huerta	74,1922	19,1232
IFSZ	57,0550	20,9579
Polini	40,8415	8,5479
Mu	48,0914	59,6708
MinAge	11,2351	3,1881
SOL	9,7934	2,8848
Crawford	5,4613	1,6190
num_adj	2,4412	1,7036
num_adv	1,7928	1,5848
num_noun	8,3726	3,0771
num_verb	5,2265	2,5698
num_dif_rare_words	4,10441	2,197713

Tabla 14. Métricas de los tweets publicados por políticos

*en las celdas con una diferencia de media mayor a un punto con respecto al resto de clases

Media y desviación estándar de las métricas calculadas más representativas (periodista)

Métrica	mean	std
LC	4,1260	2,2366
SSR	153,8122	33,7572
SCI	14,6455	10,1111
ARI	10,2686	6,0279
MeanEmdedding	6,6961	1,4614
Huerta	77,7258	24,0472
IFSZ	64,7535	24,2413
Polini	44,0715	8,4040
Mu	59,1162	72,0874
MinAge	10,0611	3,3191
SOL	8,6484	2,8244
Crawford	4,7747	1,8552
num_adj	2,1072	1,6665
num_adv	1,9367	1,6729
num_noun	7,2821	3,3854
num_verb	5,2571	2,8013
num_dif_rare_words	3,8790	2,3104

Tabla 15. Métricas de los tweets publicados por periodistas

Gráfica de las métricas (político)

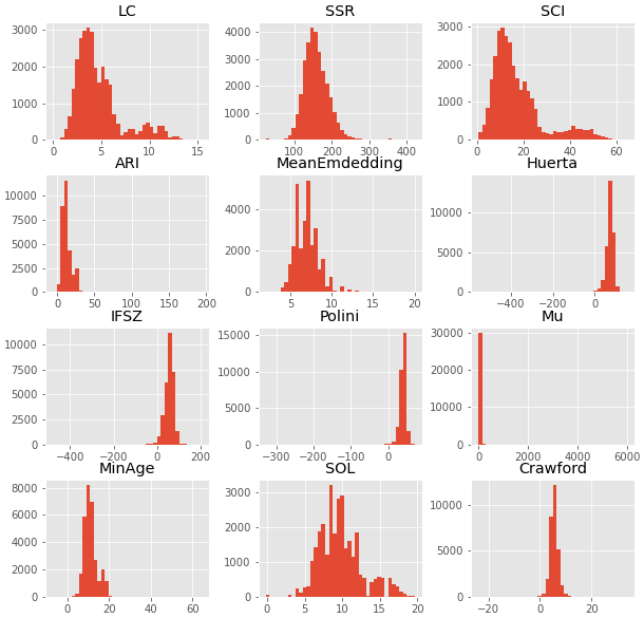

Ilustración 12. Gráficas de las métricas de los tweets de políticos

Gráfica de las métricas (periodista)

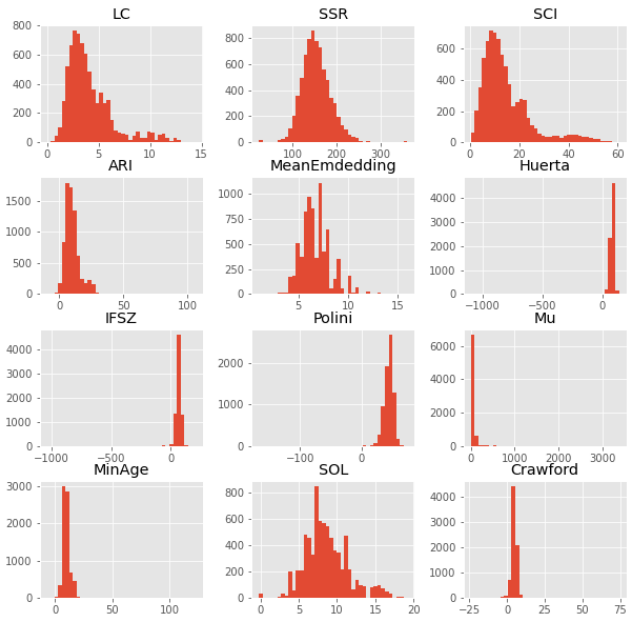

Ilustración 13. Gráficas de las métricas de los tweets de periodistas

Se puede ver que en la clasificación por profesión es donde mayor diferencia en las métricas nos encontramos de todas las clasificaciones estudiadas, por lo que sería más fácil clasificar los tweets atendiendo a las métricas de estos. Los políticos emplean más nombres, adjetivos y “palabras raras”, mientras que en general los índices de legibilidad son superiores en los periodistas, de hecho, son todos superiores salvo SSR, SCI y ARI.

3.1.4 Ideología Multiclase

En este apartado vamos a realizar el análisis de los textos de los autores según su ideología multiclase. En la *tabla 16*, donde mostramos las 10 palabras más usadas por cada tipo, se observa que las dos palabras más usadas son las mismas para todas las clases (“gobierno” y “España”), apareciendo “Sanchez” en el top 3 y “españoles” en el top 10 de los usuarios de la derecha (moderada o no) mientras que en los de la izquierda no parece. Entre la derecha modera y la no moderada la principal diferencia es que la moderada usa más “presidente” y “país” mientras que la no moderada emplea más “libertad” y “ley”. En cuanto a la izquierda, difieren en que la moderada emplea más “mujeres” y la no moderada “vida”.

	Izquierda		Izquierda Moderada		Derecha Moderada		Derecha	
	palabra	contador	palabra	contador	palabra	contador	palabra	contador
1	gobierno	2665	gobierno	1646	gobierno	2295	gobierno	2948
2	españa	2031	españa	1414	españa	1735	españa	2382
3	años	1835	años	953	sánchez	1456	sánchez	1689
4	gracias	1322	gracias	896	años	866	años	1239
5	país	1300	país	785	estado	695	españoles	1021
6	madrid	1189	estado	595	españoles	683	gracias	960
7	estado	1137	personas	565	gracias	683	estado	901
8	personas	1078	madrid	546	madrid	631	madrid	843
9	vida	936	mujeres	496	presidente	495	libertad	683
10	política	929	política	495	país	470	ley	614

Tabla 16. Palabras más usadas en los tweets de cada ideología política

La *tabla 17* contiene un resumen de la media y la desviación estándar de las métricas de los textos según la ideología política multiclase, donde destaca que la derecha moderada presenta un índice muy pequeño de "palabras raras", un 0,4. Empleando tanto los usuarios de izquierdas como los de derechas un mayor número de verbos, nombres, adjetivos, adverbios que su homónimo moderado del mismo modo que los índices de legibilidad salvo ARI, SCI y SSR donde es superior en los usuarios menos moderados (de izquierdas y de derechas).

	Izquierda		Izquierda Moderada		Derecha Moderada		Derecha	
Metrica	mean	std	mean	std	mean	std	mean	std
LC	4,5507	2,3261	4,6737	2,4696	4,6580	2,3988	4,4386	2,2777
SSR	153,2685	30,9132	159,0327	33,9068	157,8290	32,5850	157,3867	30,5186
SCI	16,5394	10,5725	16,8176	11,2395	16,9063	10,9286	15,9345	10,4123
ARI	11,4619	5,9256	12,9014	6,9354	12,2588	6,2019	11,5413	6,1481
MeanEmdedding	6,9705	1,4344	6,8214	1,4300	6,8808	1,3932	6,8740	1,3920
Huerta	77,0786	16,0950	71,6276	21,3085	75,8520	22,1626	76,7178	18,6855
IFSZ	60,9140	18,7545	55,1291	23,0726	59,2503	22,9562	61,2092	20,3762
Polini	42,5649	7,2700	40,3347	10,0290	41,4712	8,0502	42,3097	7,9895
Mu	53,4628	81,7503	49,2548	50,1661	48,0073	54,3762	51,9725	65,7297
MinAge	10,7365	3,0137	11,3844	3,3738	10,9670	3,3113	10,6414	3,1011
SOL	9,4585	2,8048	9,7377	3,0383	9,5899	2,8687	9,3015	2,8311
Crawford	5,1731	1,4833	5,5652	1,8376	5,2719	1,7204	5,1408	1,5230
num_adj	2,5409	1,7357	2,3323	1,7298	2,2907	1,6489	2,3711	1,6644
num_adv	2,0852	1,6909	1,6026	1,5369	1,8051	1,5632	1,9118	1,6106
num_noun	8,5217	3,1191	7,8815	3,2898	8,1181	3,0648	8,2654	3,1254
num_verb	5,8147	2,5812	4,5725	2,5548	5,3137	2,5432	5,6089	2,6454
num_dif_rare_words	4,2933	2,1669	3,7006	2,1708	0,4020	0,2189	4,6134	2,3565

Tabla 17. Métricas de los tweets publicados por los autores de cada ideología política

*en las celdas con una diferencia de media mayor a un punto

Así mismo se realizó un estudio para ver si existía una correlación entre las métricas y las distintas clasificaciones (*tabla 18*), pero no se encontró ninguna salvo la existente como era de esperar entre la ideología binaria y multiclase:

	gender	profession	ideology_binary	ideology_multiclass
LC	0,0266	0,0998	-0,0062	-0,0066
SSR	-0,0095	0,0494	0,0171	0,0389
SCI	0,0317	0,0916	-0,0040	-0,0081
ARI	-0,0029	0,1476	-0,0190	0,0048
MeanEmdedding	0,0349	0,0650	-0,0022	-0,0173
Huerta	0,0682	-0,0694	0,0529	0,0119
IFSZ	0,0394	-0,1400	0,0509	0,0152
Polini	0,0224	-0,1490	0,0252	-0,0056
Mu	-0,0141	-0,0701	-0,0146	-0,0164
MinAge	-0,0113	0,1436	-0,0366	-0,0129
SOL	0,0142	0,1564	-0,0198	-0,0129
Crawford	-0,0327	0,1614	-0,0485	-0,0167
num_adj	0,0245	0,0780	-0,0309	-0,0400
num_adv	0,0216	-0,0357	0,0089	-0,0287
num_dif_rare_words	0,0357	0,0403	0,0550	0,0350
num_noun	0,0309	0,1367	0,0013	-0,0232
num_verb	0,0559	-0,0047	0,0569	-0,0071
gender	1,0000	-0,1171	0,0997	0,0686
profession	-0,1171	1,0000	0,0166	0,0134
ideology_binary	0,0997	0,0166	1,0000	0,8725
ideology_multiclass	0,0686	0,0134	0,8725	1,0000

Tabla 18. Tabla de correlación de las métricas más representativas para cada tipo de clasificación
(se aplica una escala de color a los datos, más rojo cuanto el valor es más pequeño y más azul a valores más grandes)

Si bien, sí se encontraron correlaciones entre las métricas calculadas por las librerías anteriores (*tabla 19*) que tendremos en cuenta a la hora de clasificar atendiendo al valor de las métricas:

Métrica 1	Métrica 2	Correlación Pearson
SCI	LC	0.9333
SOL	LC	0.9038
MinAge	ARI	0.8847
SOL	MinAge	0.8634
SOL	SCI	0.8512
MinAge	LC	0.8069
Polini	ARI	-0.8627
Crawford	IFSZ	-0.8967
MinAge	IFSZ	-0.9087

Tabla 19. Tabla de correlación con las métricas con mayor correlación

3.2 Resumen Análisis Dataset

En este apartado hemos analizado el dataset de entrada, en el cual tenemos 120 tweets de 313 usuarios distintos lo que hace un total de 37560 tweets diferentes. El dataset presenta por fila la siguiente información:

- label: indica el usuario anonimizado de Twitter del autor del tweet
- gender: género masculino o femenino del autor
- profession: la profesión del autor, político o periodista
- ideology_binary: indica si el autor es de derechas o izquierdas
- ideology_multiclass: clasificación multiclase de la ideología política, izquierdas, izquierda moderada, derechas o derecha moderada
- tweet: texto del tweet

Se ha visto que las clases no están balanceadas para ninguna de las clases, tenemos más autores de izquierdas (un 56,87% de los casos), más de género masculino (el 56,55%), muchos más políticos (80,35%) que periodistas y más ejemplos de ideología moderada de izquierdas (32,27%) que el resto de sus respectivas clases.

Dentro de cada clase se ha estudiado que palabras utiliza cada una, así hemos visto que tanto en los textos de los autores de izquierdas y de derechas las palabras más usadas son “gobierno” y “España”, si bien presentan varias palabras diferentes dentro de su lista de palabras más empleadas. En cuanto a la clasificación por género y la por profesión sus respectivas listas de palabras más usadas son bastantes similares. Destaca que para el género femenino aparece la palabra “mujeres” en el top 10 de más usadas mientras que en el masculino aparece “política”. Mientras que en los textos de políticos aparecen las palabras “ley” y “personas” en la lista de más usadas, en los periodistas nos encontramos con “gente” y “vida.

En cuanto a las métricas la mayor diferencia se encuentra para la clasificación por profesión donde se puede ver que los políticos emplean más nombres, adjetivos y “palabras raras”, mientras que en general los índices de legibilidad son superiores en los periodistas.

4. NORMALIZACIÓN TWEETS ENTRADA

En esta fase realizamos un proceso de transformación del texto original (tweets en nuestro caso) para obtener una forma canónica, con el objetivo de uniformizar la forma del texto y facilitar con ello posteriores procesos. Para ello hemos aplicado las siguientes transformaciones, algunas de las cuales (convertir a minúsculas, eliminación de sígnos de puntuación, eliminar emojis,…) se ha observado que se han utilizado también en otros alguno de los trabajos detallados en el apartado 2:

- **Convertir texto a minúscula:** En el primer paso de la normalización convertimos todos los textos a minúscula para facilitar todos los procesos posteriores.

- **Aplicar expresiones regulares:** Las expresiones regulares son secuencias de caracteres que buscan patrones comunes y permiten efectuar operaciones de sustitución en textos. Mediante expresiones regulares hemos eliminado monosílabos y símbolos alfanuméricos (¡, !, &, signos de puntuación, ….) por ser poco informativos y con el fin de facilitar el análisis de las palabras de los textos. También hemos empleado expresiones regulares para normalizar determinadas palabras que no eran tratadas por la librería usada en la lematización, que explicaremos a continuación. Por ejemplo, "*covid19"* y "*covid 19"* se han sustituido por el término general "*covid*".

- **Eliminar stopwords:** El término stopwords se refiere a aquellas palabras sin significado propio que suelen ser las más comunes en un idioma (artículos, proposiciones, conjunciones, …). Estas palabras se suelen eliminar en muchos procesos de PLN con el fin de centrar el análisis en las palabras con contenido semántico. En un principio se utilizó la librería nltk de Python importando el diccionario de stopwords en español, pero finalmente se ha optado por emplear el diccionario de stopwords de la librería stopwordsiso por ser más completo.

- **Eliminar emojis:** Con el fin de normalizar también se han eliminado los emojis presentes en los tweets de entrada para ello se ha empleado la librería emoji.

- **Stemming / Lemmatización:** Stemming es el proceso que aplica la reducción de palabras a su base o raíz. Este tipo de preprocesado de textos hace que el vocabulario se reduzca, agrupando las palabras cuya raíz es la misma. Por ejemplo, si aplicamos stemming a las palabras "*canto", "cantas", "canta", "cantáis", "cantan"* nos quedaríamos con la raíz "*cant"*.

  La lematización es un proceso muy similar a stemming ya que reduce las palabras a una base común, pero no corta las terminaciones. Relaciona una palabra con su forma canónica o lema, entendiendo por lema la forma que tienen las palabras cuando las buscamos en el diccionario. Por ejemplo, las palabras "*canto", "cantas", "canta", "cantáis", "cantan"* son distintas conjugaciones del verbo "*cantar"*, el cual sería el lema por el que sustituirían todas ellas al realizar el proceso de lematización.

  Como hemos visto al reducir una palabra a su raíz (stemming) perdemos el significado de esta, es decir, la palabra se convierte en una palabra no válida en

el lenguaje, se ha optado por lematizar los textos de entrada de nuestro modelo. Para la lematización se ha empleado la librería *stanza* que nos permite lematizar textos en español.

5. MODELOS DE PREDICCIÓN Y CLASIFICACIÓN

En este apartado describiremos brevemente los algoritmos de clasificación empleados y cómo representamos la información de los tweets para que puedan ser empleados por dichos algoritmos.

5.1 Representación de la información

Para poder utilizar los datos de texto en los algoritmos de aprendizaje automático, necesitamos representar los textos de forma que los pueda entender y utilizar el sistema. En problemas relacionados con el PLN, se utiliza el concepto de Bag-of-Words para referirse al modelo utilizado para extraer las características de las palabras (D'Souza, 2018). Para el presente trabajo se ha utilizado:

- ***TF-IDF***: Es una medida estadística utilizada para evaluar cómo de importante es una palabra para un documento en una colección o corpus. La importancia aumenta proporcionalmente al número de veces que aparece una palabra en el documento, pero se compensa con la frecuencia de la palabra en el corpus, lo que permite manejar el hecho de que algunas palabras son generalmente más comunes por ello y por ser utilizada a menudo en tareas de minería de textos nos hemos decidido a emplearla en nuestro trabajo.

  Se calcula como el producto de dos términos:

  Tfidf (t,d,D) = tf(t,d) x idf(t,D)

  Donde:

  t: termino
  d: documento
  D: nº total documentos
  tf**:** (Term Frequency) mide la frecuencia con la que aparece un término en un documento. Dado que cada documento tiene una longitud diferente, es posible que un término aparezca muchas más veces en documentos largos que en documentos cortos. A continuación, se muestran los diferentes cálculos de peso que se pueden utilizar para esta medida:

weighting scheme	tf weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$
log normalization	$\log(1 + f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K)\frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Ilustración 14. Variantes del cálculo del peso frecuencia de término (tf)

idf**:** (Inverse Document Frequency), mide la importancia de un término. Al calcular TF, todos los términos se consideran igualmente importantes, necesitamos ponderar los términos frecuentes y escalar los raros. En la siguiente tabla se muestran los diferentes cálculos de peso que se pueden utilizar para esta medida:

weighting scheme	idf weight ($n_t = \|\{d \in D : t \in d\}\|$)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log\left(\frac{N}{1+n_t}\right)+1$
inverse document frequency max	$\log\left(\frac{\max_{\{t' \in d\}} n_{t'}}{1+n_t}\right)$
probabilistic inverse document frequency	$\log \frac{N-n_t}{n_t}$

Ilustración 15. Variantes del cálculo del peso frecuencia de documento inversa (idf)

- ***Word Embeddings***: Cada palabra está representada por un vector teniendo en cuenta su ocurrencia y la información de coocurrencia (relación de proximidad de dos o más términos en una unidad de texto) (Naryani, 2017). En Word Embeddings las palabras individuales se representan como vectores de valores reales en un espacio vectorial predefinido y donde las palabras que tienen el mismo significado tienen una representación similar (Brownlee, 2017a). Cada palabra se asigna a un vector y los valores del vector se aprenden. Word Embeddings aprende en función del uso y contexto de las palabras lo que permite que las palabras que se usan de manera similar den como resultado una representación similar, con el fin de capturar de forma natural su significado. Cómo método para implementar Word Embedding existen varias alternativas como “word2vec” o GloVe (Global Vector for Word Representation). En nuestro caso emplearemos Word2Vec por haber sido estudiado durante el master y porque en teoría requiere un menor costo de memoria que Glove (ejecutaremos nuestros algoritmos en instancias de Google Colab con memoria “limitada”), ya que este último utiliza una matriz de co-ocurrencia e información global.

  Word2vec es una herramienta para entrenar/generar y usar word embeddings que utiliza modelos predictivos, aprende el significado de las palabras procesando un gran corpus de texto sin etiquetas. Fue desarrollado por un equipo de investigadores dirigido por Tomas Mikolov en Google [10] [11].

  Word2vec utiliza una red neuronal de 2 capas que toma como entrada un corpus textual y genera sus correspondientes word embeddings que pueden ser la entrada de una red de aprendizaje profundo o servir para detectar relaciones entre palabras.

Word2Vec implementa dos modelos neuronales:

- *Modelo skip-gram* el modelo predice las palabras contexto a partir de la palabra objetivo
- *Modelo bolsa de palabras* continua (CBOW) el modelo predice la palabra objetivo dada una ventana de palabras (el contexto)

En general el modelo skip-gram obtiene la precisión más alta mientras que CBOW es menos costoso computacionalmente y produce resultados con precisión similar.

Para algunos algoritmos, por ejemplo, BERT, se le ha pasado directamente los textos normalizados de entrada y también se ha estudiado si era posible emplear las métricas calculadas en la fase de análisis para la clasificación de los tweets.

5.2 Algoritmos de Clasificación

Una vez analizados y normalizados los datos de entrada se han empleado una serie de algoritmos de clasificación para poder evaluar con cual obtenemos los mejores resultados.

Se ha querido utilizar una representación de los algoritmos empleados durante todo el Master:

- Regresión Logística
- SVM
- Random Forest
- LightGBM
- MLP
- Red neuronal
- Red Neuronal con capa Word2Vec preentrenada
- BERT

A continuación, describiremos brevemente cada uno de ellos.

5.2.1 Regresión Logística

El clasificador de regresión logística lineal es un modelo de clasificación supervisada basado en el modelo lineal. Al igual que un modelo de regresión lineal, un modelo de regresión logística calcula una suma ponderada de las características de entrada (más un término de sesgo), pero en lugar de generar el resultado directamente como lo hace el modelo de regresión lineal, genera la logística de este resultado.

$$\hat{p} = h_{\theta}(\mathbf{x}) = \sigma(\theta^{\top}\mathbf{x})$$

Probabilidad estimada del modelo de regresión logística

Para problemas de clasificación binaria, se utiliza la función sigmoide que toma valores entre 0 y 1:

$$\sigma(t) = \frac{1}{1 + \exp(-t)}$$

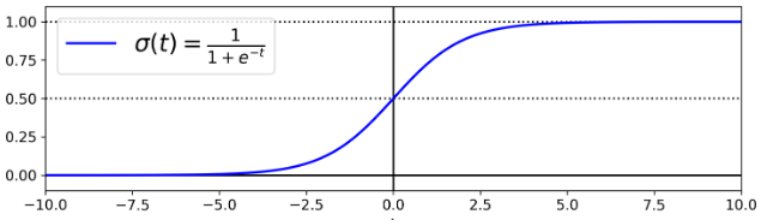

Ilustración 16. Gráfica función sigmoide

La idea en este tipo de problemas binarios, donde las variables de entrada toman valores continuos entre 0 y 1, pero la variable de salida toma valores discretos (0 o 1), es establecer un punto de corte denominado threshold. De manera que los valores que se encuentran por encima de este punto de corte pertenecen a la clase positiva (1), mientras que valores por debajo de dicho punto de corte se clasifican en la clase negativa (0).

Mientras que, en problemas de clasificación con varias clases, la función que se utiliza se denomina softmax, dicha función calcula las probabilidades de cada clase sobre todas las clases posibles (predice la clase con la probabilidad estimada más alta). Estas probabilidades determinan la predicción de la variable categórica según los datos de entrada:

$$\hat{p}_k = \sigma(\mathbf{s}(\mathbf{x}))_k = \frac{\exp(s_k(\mathbf{x}))}{\sum_{j=1}^{K} \exp(s_j(\mathbf{x}))}$$

Donde:

- *K* es el número de clases.
- s(x) es un vector que contiene las puntuaciones de cada clase para la instancia x.
- *σ* (s(x))*k* es la probabilidad estimada de que la instancia x pertenezca a la clase *k*, dados las puntuaciones de cada clase para esa instancia.

Para este método se ha realizado afinación de los siguientes hiperparámetros:

- *solver*: indicamos el algoritmo a utilizar en el problema de optimización, por defecto se emplea 'lbfgs'. En general para conjuntos de datos pequeños, 'liblinear' es una buena opción, mientras que 'sag' y 'saga' son más rápidos para conjuntos de entrada de mayor tamaño. Los posibles valores que tendremos en cuenta para la afinación del hiperparámetro solver serán los siguientes: *'newton-cg', lbfgs', 'liblinear', 'sag', 'saga'*)

- *penalty*: establecemos el tipo de regularización:
  - *L1*: hace que algunas β sean 0 siendo una forma de forzar la selección de variable
  - *L2*: también conocida como Ridge, lo que hace es estimar β pequeños, sirve para controlar el sobre ajuste.

- *elasticnet*: sirve para aplicar la regularización L1 y L2 a la vez, hay que fijar un float entre 0 y 1, siendo 0 la aplicación completa de la regularización L1 y nada de la L2 y viceversa.

- *c*: parámetro que aplica regularización con el objetivo de reducir el overfitting, cuanto más pequeño es el valor, mayor es la regularización. Consideraremos los siguientes valores para la afinación de "c": 0.1, 1, 10, 100 y 1000

5.2.2 SVM

Las máquinas de vectores de soporte (*support-vector machines*, SVM) tienen su origen en los trabajos sobre la teoría del aprendizaje estadístico y fueron introducidas en los años 90 por Vapnik y sus colaboradores [Boser et al.,1992, Cortes & Vapnik,1995]. Las SVM pertenecen a la categoría de los clasificadores lineales, tratan de predecir las categorías a las que pertenecen los datos a través de una función lineal conocida como hiperplano.

Una SVM selecciona un hiperplano de separación que equidista de los ejemplos más cercanos de cada clase para conseguir un margen máximo a cada lado del hiperplano (m en la figura). Además, a la hora de definir el hiperplano, sólo se consideran los ejemplos de entrenamiento de cada clase que caen justo en la frontera de dichos márgenes. Estos ejemplos reciben el nombre de vectores soporte.

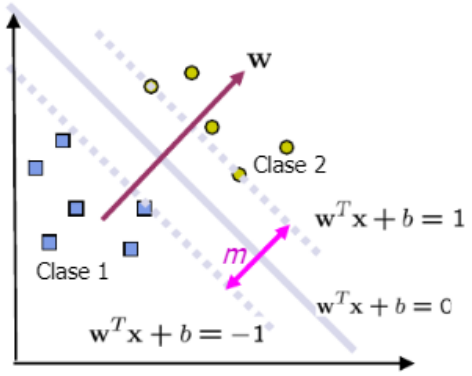

Ilustración 17. Ejemplo funcionamiento SVM
(fuente https://www.researchgate.net/publication/49588125_LAS_MAQUINAS_DE_SOPORTE_VECTORIAL_SVMs)

En la práctica, los datos a menudo no pueden ser linealmente separables en el espacio de características de entrada, en estos casos se recurre a espacios de dimensión mayor donde los puntos sí son linealmente separables y se requiere de una función conocida como kernel trick:

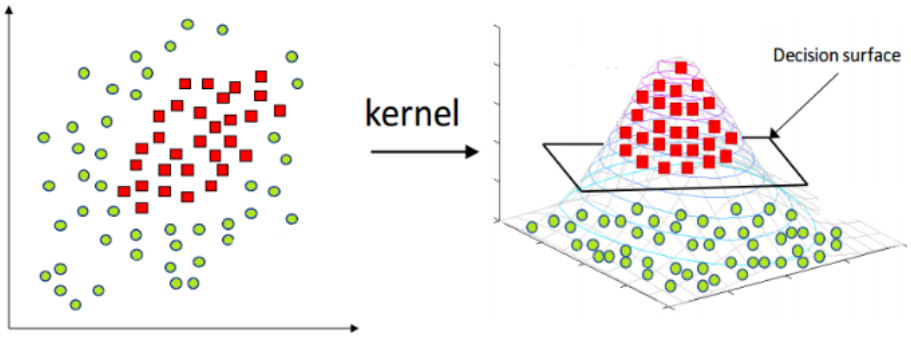

Ilustración 18. Ejemplo ilustrativo del "kernel trick"
(fuente https://medium.com/@zxr.nju/what-is-the-kernel-trick-why-is-it-important-98a98db0961d)

Para mejorar la precisión del modelo, ajustamos los siguientes hiperparámetros:

- *kernel*: la función principal del núcleo es transformar un espacio de entrada de baja dimensión en un espacio de mayor dimensión. Es principalmente útil en problemas de separación no lineal. Se han considerado las siguientes opciones: *'rbf'*, *'poly'* y '*sigmoid':*

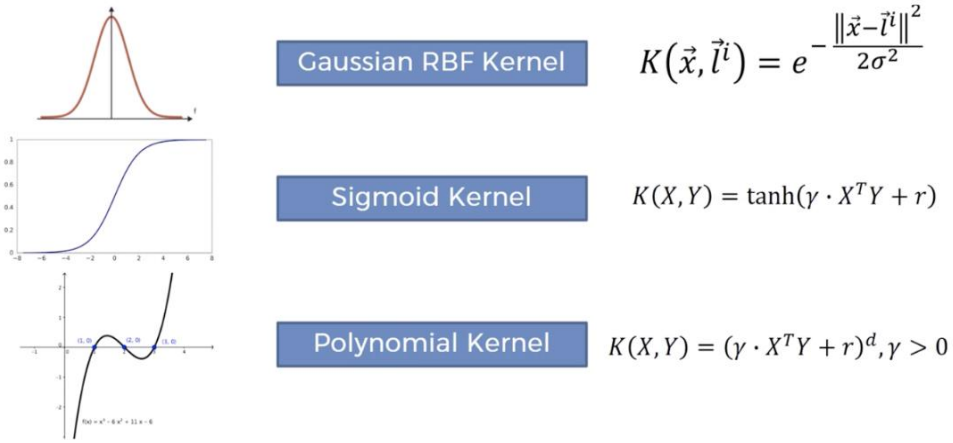

Ilustración 19. Tipos de Kernel SVM (fuente [HTTP15])

- *C (Regularización)*: indica la penalización que representa la clasificación errónea. El término de clasificación errónea le dice a la optimización de SVM cuánto error es soportable, permitiendo controlar el sobreajuste. Tendremos en cuenta los siguientes posibles valores de C: 0.1, 1, 10 y 100.

- *Gamma*: define cuánto influye cada dato en el cálculo de la línea de separación, para valores altos de gamma, los puntos cercanos tendrán una gran influencia mientras que si gama es baja también se consideran los puntos lejanos para obtener el límite de decisión:

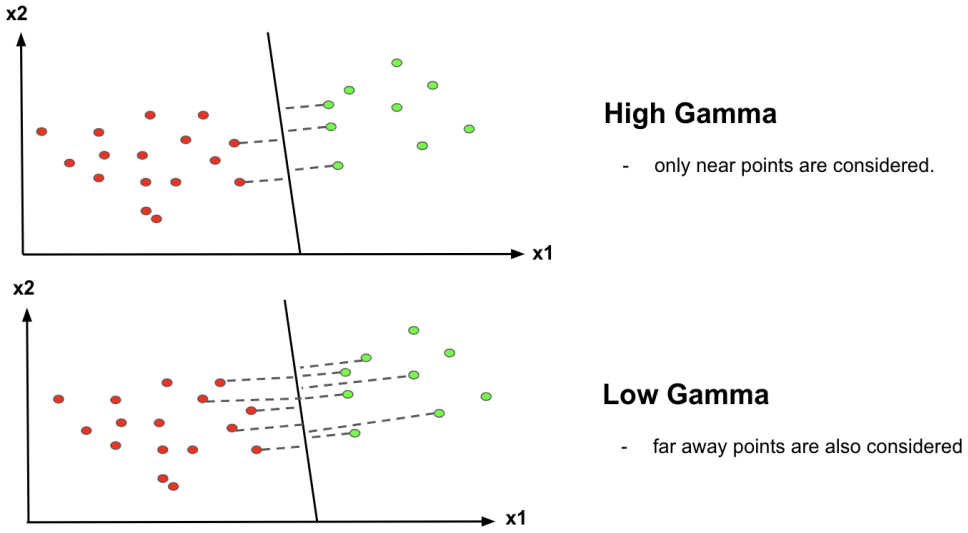

Ilustración 20. Ejemplo ilustrativo del comportamiento de Gamma (fuente [HTTP15])

Valores considerados para gamma a la hora de la afinación: 1, 0.1, 0.01 y 0.001

5.2.3 Random Forest

Random forest es un algoritmo de aprendizaje automático que combina la salida de múltiples árboles de decisión para llegar a un único resultado. El algoritmo es una extensión del método de "bagging", utiliza tanto el "bagging" como la aleatoriedad de características para crear un bosque de árboles de decisión no correlacionado. La aleatoriedad de características genera un subconjunto aleatorio de características, lo que garantiza una baja correlación entre los árboles de decisión. Esta es una diferencia clave entre los árboles de decisión y los bosques aleatorios. Mientras que los árboles de decisión consideran todas las divisiones de características posibles, los bosques aleatorios solo seleccionan un subconjunto de esas características

El algoritmo de bosque aleatorio se compone de una colección de árboles de decisión, y cada árbol del conjunto se compone de una muestra de datos extraída de un conjunto de entrenamiento con reemplazo. De esa muestra de entrenamiento, un tercio se reserva como datos de prueba, conocidos como la muestra "out-of-bag" (oob). Después, se inyecta otra instancia de aleatoriedad a través del empaquetamiento de características, lo que agrega más diversidad al conjunto de datos y reduce la correlación entre los árboles de decisión. Finalmente, en una tarea de clasificación, un voto mayoritario (la variable categórica más frecuente) producirá la clase predicha. La muestra oob se usa para la validación cruzada.

Entre los distintos hiperparámetros a ajustar se han considerado los siguientes:

- *max_depth*, la profundidad máxima del árbol, por defecto, los nodos se expanden hasta que todas las hojas son puras o hasta que todas las hojas contengan menos que el valor "min_samples_split". Consideraremos los siguientes valores a la hora de afinar este parámetro: 5,10,25 y 50

- *n_estimators*, número de árboles de decisión que se generarán. Utilizaremos los siguientes valores para su afinación: 25, 50, 100 y 200

- *max_features*, cantidad de características a considerar cuando se busca la mejor división de los datos. Tendremos en cuenta las siguientes opciones a la hora de optimizarlo: *'auto', 'sqrt', 'log2'* y *None*

5.2.4 LGBM Classifier

LightGBM (Light Gradient Boosted Machine) utiliza una técnica de gradiente de muestreo unilateral (GOSS) en la que se toman instancias de los datos con mayores gradientes en cada iteración y un porcentaje de muestreo aleatorio del resto de datos, con esto se consigue que el entrenamiento y el ajuste de hiperparámetros sea rápido. Mientras que los árboles de otros algoritmos crecen horizontalmente, el algoritmo LightGBM crece verticalmente, es decir crece por hojas en lugar de por niveles, eligiendo la hoja con mayor pérdida para crecer.

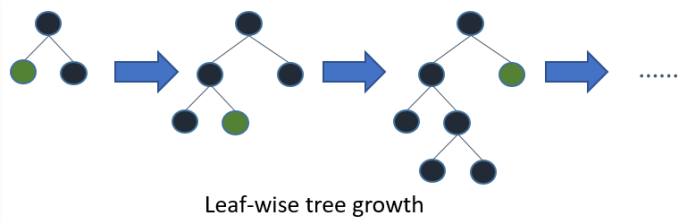

Ilustración 21. Ejemplo crecimiento por hojas del algoritmo LightGBM (fuente [HTTP8])

Hemos afinado los siguientes hiperparámetros:

- *max_depth*, profundidad máxima que pueden alcanzar los árboles, con los siguientes valores: 5, 10, 25 y 50

- *n_estimators,* número de árboles que se generarán, tendremos en cuenta estos valores para su afinación: 25, 50, 100 y 200

- *learning_rate*, tasa de aprendizaje, cuyos valores considerados son 0.0001, 0.001, 0.01 y 0.1

5.2.5 Red Neuronal

Una red neuronal artificial (RNA) es un modelo computacional que consiste en un conjunto de unidades, llamadas neuronas, conectadas entre sí para transmitirse señales. La información de entrada atraviesa la red neuronal, se opera con ella y se producen unos valores de salida. Funciona simultaneando un número elevado de unidades de procesamiento interconectadas que parecen versiones abstractas de neuronas. Están compuestas por cuatro elementos básicos: las neuronas o nodos, las conexiones, los pesos sinápticos y las funciones de activación. La distribución de neuronas dentro de la red se realiza formando niveles o capas, con un número determinado de neuronas en cada capa. A partir de su situación dentro de la red, se pueden distinguir tres tipos de capas:

- De entrada: es la capa que recibe directamente los datos de entrada
- Ocultas: son internas a la red y no tienen contacto directo con el entorno exterior. Las neuronas de las capas ocultas pueden estar interconectadas de distintas maneras, lo que determina, junto con su número, las distintas topologías de redes neuronales
- De salida: transfieren información de la red hacia el exterior.

Las reglas que rigen el funcionamiento y la estructura de una RNA son:

I. El procesamiento de información se produce en muchos elementos simples denominados nodos, células o neuronas.
II. Las señales se pasan entre nodos a través de enlaces de conexión.
III. Cada enlace de conexión tiene un peso asociado que representa su fuerza de conexión.
IV. Cada nodo aplica una transformación no lineal llamando una función de activación a su entrada de red para determinar su señal de salida

Para las redes neuronales hemos afinado los siguientes hiperparámetros:

- *learn_rate*, la tasa de aprendizaje del optimizador controla el tamaño de paso para que un modelo alcance la función de pérdida mínima. Una tasa de aprendizaje alta hace que el modelo aprenda más rápido, pero puede alcanzar un óptimo local. Una tasa de aprendizaje más baja da una mejor oportunidad de encontrar una función de pérdida mínima. a consta de más recursos, tiempo y capacidad de memoria. Se han empleado estos valores para su afinación:0.0001, 0.001, 0.01 y 0.1

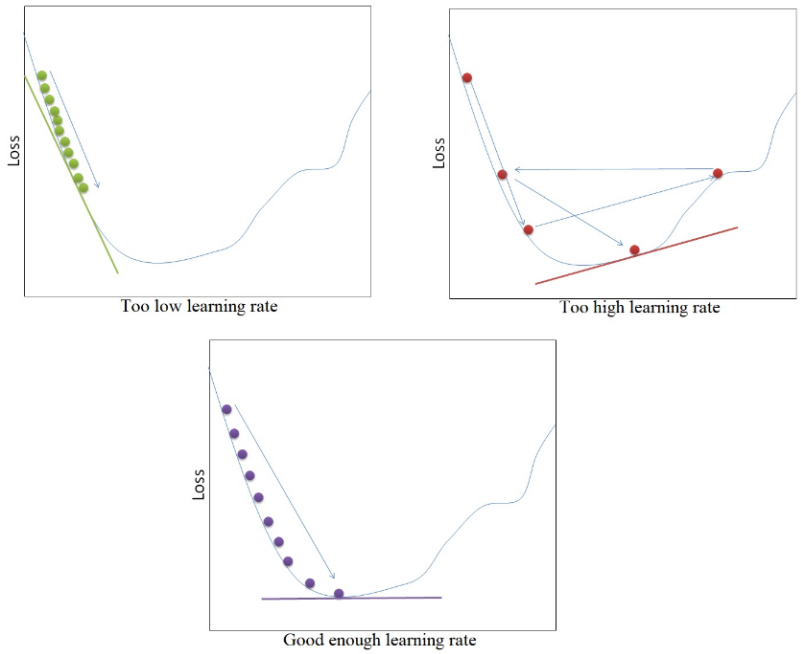

Ilustración 22. Ejemplo del comportamiento de los modelos en función de la tasa de aprendizaje (fuente [HTTP10])

- *epochs*, el número de veces que el modelo es expuesto al conjunto de datos de entrenamiento. Un número demasiado pequeño de epochs dará como resultado

un ajuste insuficiente porque la red neuronal no ha aprendido lo suficiente mientras que demasiadas epochs darán lugar a un sobreajuste. Utilizaremos estos valores por defectos para la afinación de epochs: 10, 50, 100 y 150

Después de varias pruebas con diferentes esquemas (número de capas, neuronas,…) y su correspondiente evaluación se optó por definir la red neuronal siguiendo el esquema que se muestra en la Ilustración 23. Empleamos como función de pérdida ”binary_crossentropy” y activación softmax si es clasificación binaria y "categorical_crossentropy" con activación sigmoid en el caso de la ideología multiclase:

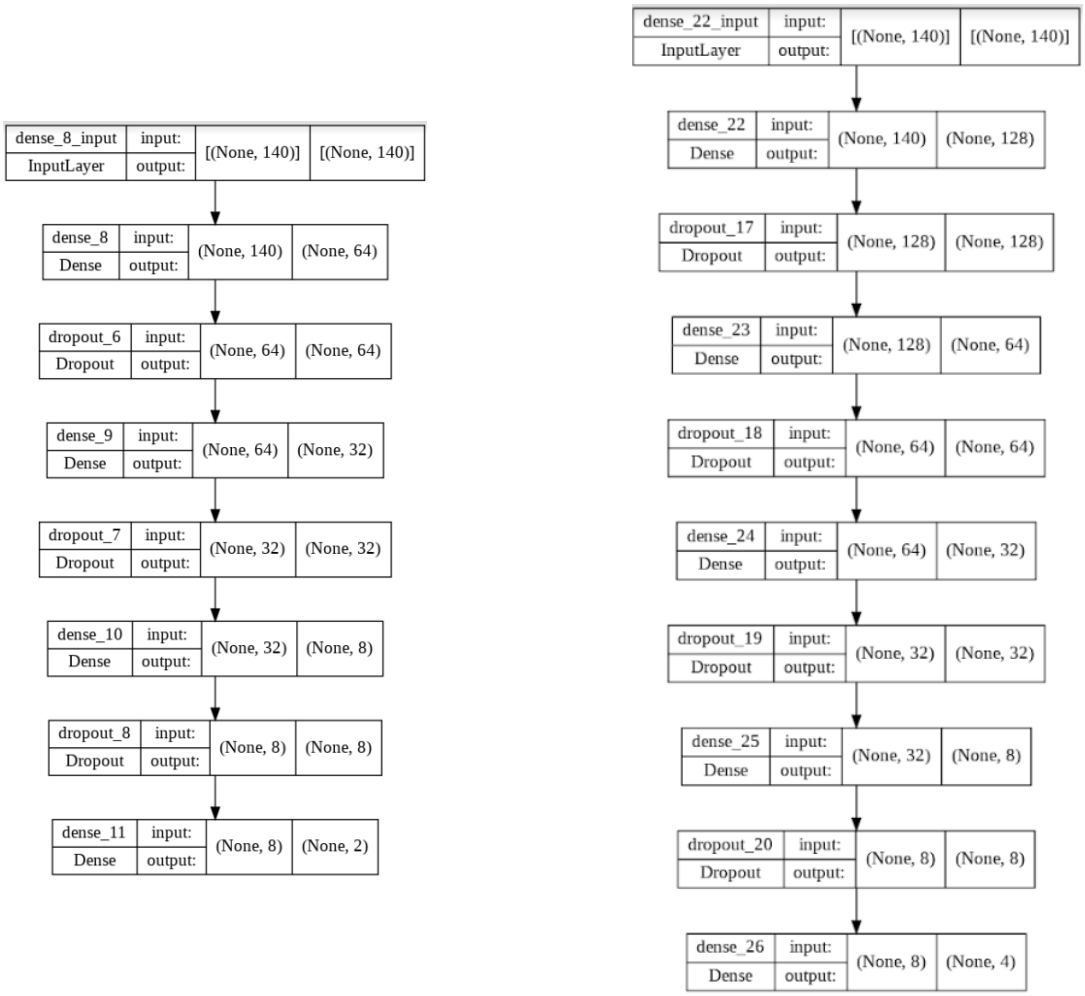

Ilustración 23. Esquema de las redes neuronales definidas para las clasisificaciones del presente TFM (a la izquierda clasificación binaria, a la derecha clasificación multilabel)

5.2.6 MLP Classifier

El perceptrón multicapa (MLP) es un algoritmo de aprendizaje supervisado que aprende una función f(x): $R^m \rightarrow R^o$, entrenando sobre un conjunto de datos, donde “*m*” es el número de dimensiones de los datos de entrada y “*o*” es el número de dimensiones para la salida. Entre la capa de entrada y la de salida puede haber una o más capas no lineales, llamadas capas ocultas donde cada capa está completamente conectada a la siguiente. Los nodos de las capas son neuronas con funciones de activación no lineales, excepto los nodos de la capa de entrada que consta de un conjunto de

neuronas $\{x_i|x_1,x_2,...,x_m\}$ que representan las entidades de entrada. Cada neurona en la capa oculta transforma los valores de la capa anterior con una suma lineal ponderada $w_1x_1+w_2x_2+...+w_mx_m$, seguida de la función de activación no lineal. La capa de salida recibe los valores de la última capa oculta y los transforma en los valores de salida.

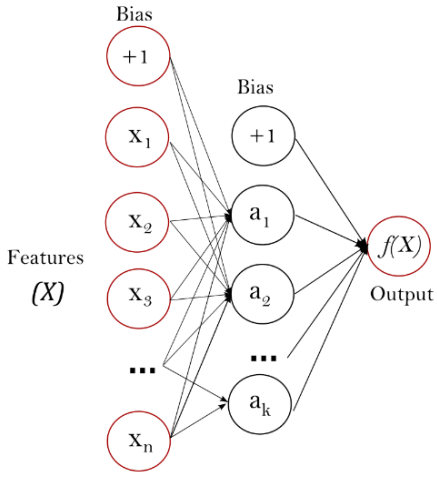

Ilustración 24. Ejemplo de MLP con una capa oculta (fuente [HTTP7])

- *max_iter*, determina el número máximo de epochs, cuyos valores posibles evaluados son: `50,100 y200`

- *alpha*, determina la penalización L2 a aplicar sobre los pesos que ponderan los valores de entrada de las neuronas. Un valor de alpha más elevado supondrá pesos menores y una menor varianza (frecuente en entornos de sobreentrenamiento). Se tuvieron en cuenta los siguientes valores de alpha: 0.1, 0.01, 0.001 y 0.0001

- *hidden_layer_sizes*, con el que establecemos el número de capas y el número de neuronas por capa. Se consideraron las siguientes tuplas para su afinación: (32,8), (64,32,8) y (128,64,32,8).

5.2.7 Transformers

El modelo *Transformer* [Polosukhin et al. 2017] apareció en 2017 y la arquitectura se dicho modelo consiste en una secuencia de *encoders* y *decoders*, a través de los cuales van fluyendo representaciones intermedias del texto de entrada, hasta producir el texto de salida. Los *encoders* y *decoders* son redes neuronales con una capa especial, denominada de autoatención, que permite determinar qué partes de la frase son las más relevantes y aprender relaciones de contexto entre las palabras. Gracias a las diferentes capas de autoatención, el procesamiento del texto se realiza de forma independiente al orden de las palabras, lo que supone un mecanismo muy potente para resolver fenómenos lingüísticos complejos como, por ejemplo, la correferencia. A finales de 2019 Google publicó T5, acrónimo de "*Text-to-Text Transfer Transformer*", un *framework* basado en *transformers* preparado para adaptar cualquier tarea de PLN y resolverla desde la perspectiva *text-to-text* de los *transformers*. Como ejemplos de sistemas basados en transformes tenemos: BERT (Bidirectional Encoder Representations from

Transformers) y GPT (Generative Pretrained Transformer). En este trabajo hemos optado por utilizar BERT por haberse estudiado durante el presente Master y porque se puede utilizar empleando la biblioteca Transformers que admite el uso de modelos de Tensorflow o Pytorch mientras que la última versión de GPT (GPT-3) es solo accesible a través de un API de OpenAI.

5.2.7.1 BERT

BERT (Bidirectional Encoder Representations from Transformers) es un modelo desarrollado por Google, el cual se puede ajustar con solo una capa de salida adicional para crear modelos de última generación para una amplia variedad de tareas, como responder preguntas e inferencia de lenguaje. BERT lee toda la secuencia de palabras a la vez, es bidireccional, analizando las palabras que se encuentran tanto a la izquierda como a la derecha de una palabra clave. Esta característica permite al modelo aprender el contexto de una palabra basándose en todo su entorno (izquierda y derecha de la palabra).

BERT tiene dos pasos principales: *pre-training* y fine-tuning. Durante el *pre-training*, el modelo se entrena con datos no etiquetados en diferentes. Mientras que en la fase de fine-tuning, el modelo BERT se inicializa primero con los parámetros pre-entrenados, y estos parámetros se ajustan con datos etiquetados de tareas posteriores. En cada tarea posterior se crean modelos ajustados separados, que se inicializan con los mismos parámetros del entrenamiento previo, pre-training.

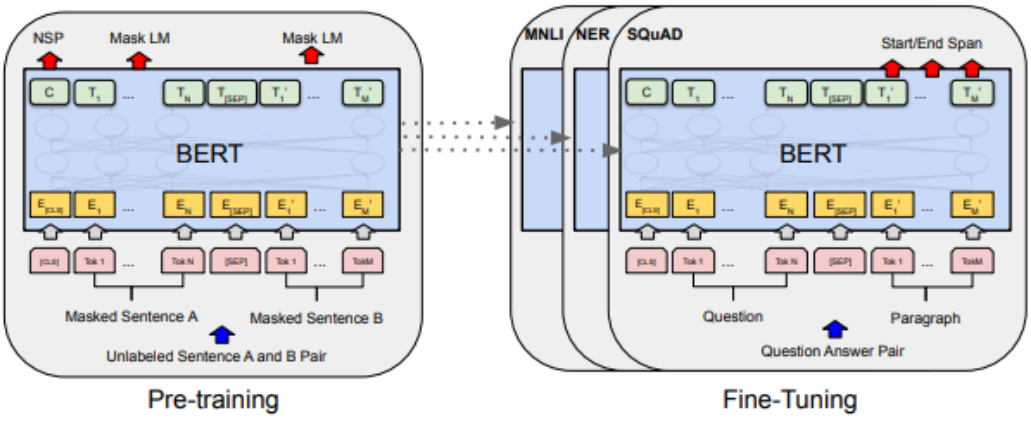

Ilustración 25. Arquitectura general de pre-training y fine-tuning del algoritmo BERT. (Fuente [9])

BERT puede utilizarse para una gran variedad de tareas lingüísticas (clasificación de textos, generación de resúmenes, respuestas a preguntas, …), añadiendo sólo una pequeña capa al modelo central. Para la clasificación, la tarea realizada en el presente trabajo, se añade una capa de clasificación sobre la salida del transformador.

En este caso se ha empleado la tasa de aprendizaje (5e-5, 3e-5, 2e-5) y el optimizador (Adamw) recomendados por Google y se ha probado con diferentes 'epochs': 10, 50 y 100.

Se han considerado dos modelos preentrenados uno de la Universidad de Chile denominado BETO, en español, y otro multi idioma, "bert-base-multilingual" que incluía el

español. Al final se optó por el segundo de ellos por obtener unos resultados ligeramente mejores en nuestro caso de uso.

6. EVALUACIÓN DE RESULTADOS

Para evaluar los resultados obtenidos por los diferentes métodos de clasificación se ha partido del dataset de evaluación publicado en la competición "IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology". En dicho dataset tenemos datos de 105 autores con 120 tweets por autor, en total12.600 tweets.

Se han considerado principalmente tres métricas para evaluar y entender los resultados: la matriz de confusión que nos ayudará a ver de forma explícita cuándo una clase es confundida con otra, la "accuracy" con la que veremos el porcentaje total de valores correctamente clasificados y f1-score que es la métrica empleada en la competición a la hora de clasificar los distintos trabajos y además es más idónea que la accuracy al tratarse de datos no balanceados por lo que será la métrica principal que consideraremos.

- **matriz de confusión**: para una tarea de clasificación de N clases es una matriz de NxN donde la celda(x,y) contiene el número de veces que un elemento con la clasificación correcta 'x' fue clasificado por el modelo como 'y'. Nos permite ilustrar el rendimiento del clasificador en función de los verdaderos positivos (TP), falsos positivos (FP), verdaderos negativos (TN) y falsos negativos (FN):

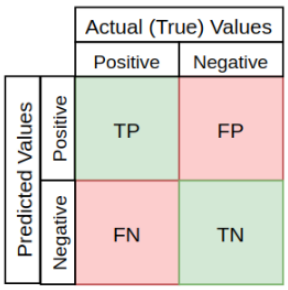

Ilustración 26. Definición de los valores representados en una matriz de confusión de un problema de clasificación binario

En la ilustración 27 mostramos un ejemplo de matriz de confusión donde la clase '1' se ha clasificado 70 veces de forma correcta, 10 veces de forma errónea como clase '2', 15 de forma errónea como clase '3' y 5 de forma errónea como clase '4'.

Valores predichos

Valores correctos	Clase	1	2	3	4
	1	70	10	15	5
	2	8	67	20	5
	3	0	11	88	1
	4	4	10	14	72

Ilustración 27. Ejemplo matriz confusión

- **accuracy**: mide el porcentaje de casos que el modelo ha acertado, cuya fórmula es la siguiente:

$$a = \frac{tp + tn}{tp + tn + fp + fn}$$

Donde

- tp: verdaderos positivos
- tn: verdaderos negativos
- fp: falsos positivos
- fn: falsos negativos

Para datos no balanceados, como es nuestro caso, no es una buena métrica para evaluar los algoritmos. Por ejemplo, si consideramos nuestro dataset de entrada, donde en la clasificación por profesión el 80% de los datos se corresponden con políticos, un algoritmo cuya predicción siempre sea "político" tendrá una accuracy de 0,80 lo cual es un valor bastante alto, pero nunca será capaz de predecir la clase "periodista".

- **f1-score**: El f1-score (valor-f) se utiliza para combinar las medidas de precisión y cobertura en un sólo valor. Un valor alto indica que tenemos una precisión y una cobertura alta en nuestro sistema. Se calcula haciendo la media armónica entre la precisión y la cobertura:

$$valor - f = \frac{2 * p * r}{p + r}$$

Donde

- p: precisión
- r: cobertura

Donde la precisión se corresponde con la fracción de predicciones del modelo propuesto acertadas (coinciden con los datos de referencia), cuya fórmula sería:

$$p = \frac{tp}{tp + fp}$$

Donde

- tp: verdaderos positivos
- fp: falsos positivos

Mientras que la cobertura se corresponde con la fracción de los datos de referencia que han sido propuestas por el modelo evaluado:

$$r = \frac{tp}{tp + fn}$$

Donde

- tp: verdaderos positivos
- fp: falsos positivos
- fn: falsos negativos

F1-score al combinar precisión y cobertura, tiene en cuenta la calidad de la predicción, ¿qué porcentaje de los que hemos dicho que son la clase positiva, en realidad lo son?, y la cantidad, ¿qué porcentaje de la clase positiva hemos sido capaces de identificar?, con lo que se evita el problema que teníamos con "accuracy" en datos no balanceados.

6.1 Resultados obtenidos a nivel de autor

En este apartado mostraremos los resultados obtenidos a partir de los tweets agrupados por autor para cada uno de los modelos indicados con anterioridad, ordenando de mayor a menor f1-score de test obtenido para cada caso. Se incluye también los resultados obtenidos por el baseline suministrado en la competición, en el cual se emplean cuatro modelos basados en regresión logística con un modelo simple de bolsa de palabras (BoW) para cada clase (género, profesión, ideologia_binaria e ideologia_multiclase).

6.1.1 Ideología Binaria

En este apartado vamos a abordar la evaluación de los resultados obtenidos en la clasificación de ideología política binaria, que se pueden ver en la *tabla 20*.

Modelo	Datos Entrada	f1_scores Test	Accuracy Test
Red Neuronal	TDIDF	0,931968	0,942857
MLPClassifier	TDIDF	0,922222	0,92381
RandomForest	TDIDF	0,913151	0,914286
LogisticRegression	TDIDF	0,902451	0,913333
LGBMClassifier	TDIDF	0,893342	0,895238
SVM	TDIDF	0,861111	0,866667
BERT	TWEET	0,619593	0,590476
Red Neuronal Word2Vec	TWEET_LEMA	0,580729	0,561905
BASELINE	BOW	0.548872	0.550000
MLPClassifier	METRICAS	0,370719	0,466667
LGBMClassifier	METRICAS	0,351852	0,542857
RandomForest	METRICAS	0,351852	0,542857
LogisticRegression	METRICAS	0,328207	0,457143
Red Neuronal	METRICAS	0,328207	0,457143
SVM	METRICAS	0,313725	0,457143

Tabla 20. Resultados clasificación ideología binaría, tweets agrupados por autor

(Se aplica una escala de color a los datos, más rojo cuanto el valor es más pequeño y más azul a valores más grandes. En gris los resultados de la baseline)

Para la clasificación de ideología política binaria llegamos a alcanzar un f1-score de 0,93 así como una accuracy del 0,94 empleando la red neuronal secuencial definida por nosotros y tomando como entrada el vector TDIDF, siendo este el algoritmo que mejor se comporta. En la ilustración 28, vemos la matriz de confusión obtenida con dicho algoritmo, donde se observa que la precisión es mayor cuando clasificamos la ideología de derechas que la de izquierdas:

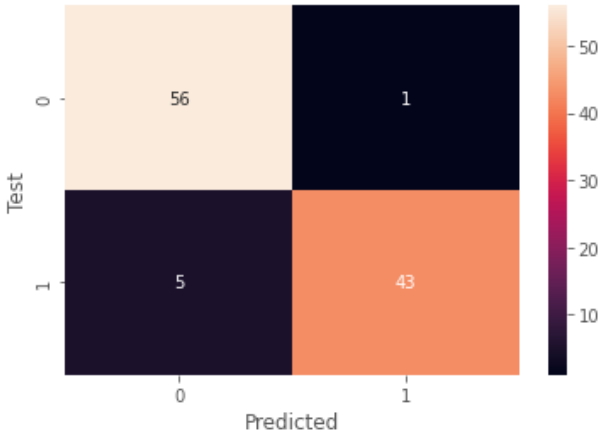

Ilustración 28. Matriz confusión red neuronal empleando TDIDF, ideología binaria (0-izquierdas, 1-derechas)

También señalar los buenos resultados obtenidos con MLPClassifier, RandomForest y LogisticRegression donde alcanzamos un f1-score y accuracy por encima de 0,9. Por el lado contrario, observamos que no obtenemos buenos resultados si partimos exclusivamente de las métricas de los textos, algo que observamos también en el resto de las clasificaciones a nivel de autor. También hay que destacar los discretos resultados obtenidos empleando Word2Vec y BERT, pero por encima del baseline. En el caso de BERT parece deberse a la limitación de tamaño de 512 tokens que presenta y en este caso al estar todos los textos escritos por un autor agrupados no se está considerando una cantidad considerable de texto de cada autor.

6.1.2 Género

A continuación, evaluaremos los resultados, *tabla 21*, obtenidos en la clasificación del género de los autores.

Modelo	Datos Entrada	f1_scores Test	Accuracy Test
LGBMClassifier	TDIDF	0,773174	0,790476
LogisticRegression	TDIDF	0,764183	0,780952
RandomForest	TDIDF	0,749603	0,771429
SVM	TDIDF	0,743767	0,780952
Red Neuronal Word2Vec	TWEET_LEMA	0,698785	0,657143
Red Neuronal	TDIDF	0,690657	0,733333
BERT	TWEET	0,678819	0,657143
MLPClassifier	TDIDF	0,666137	0,695238
Red Neuronal	METRICAS	0,503636	0,504762
BASELINE	BOW	0.466667	0.600000
LogisticRegression	METRICAS	0,396552	0,657143
SVM	METRICAS	0,389535	0,638095
LGBMClassifier	METRICAS	0,255319	0,342857
MLPClassifier	METRICAS	0,255319	0,342857
RandomForest	METRICAS	0,255319	0,342857

Tabla 21. Resultados clasificación por género, tweets agrupados por autor

(Se aplica una escala de color a los datos, más rojo cuanto el valor es más pequeño y más azul a valores más grandes. En gris los resultados de la baseline)

En general es donde obtenemos perores resultados, pero aun así con la mayoría de los métodos utilizados se obtienen mejores resultados que con el baseline de la competición. Se sigue observando que los peores resultados son si se tienen en cuenta las métricas calculadas por las librerías MultiAzterTest y TextComplexityFreeling. En esta ocasión empleando BERT y Word2Vec obtenemos unos resultados un poco mejores que en el caso anterior. Además, destaca que los dos algoritmos que mejor se comportaban para la ideología binaría en este caso son casi los que peores resultados obtienen si descartamos los algoritmos que toman como entrada las métricas.

En cuanto a los mejores resultados los obtenemos con LGBMClassifier empleando TDIDF donde llegamos a un 0,77 en el f1-score y a una accuracy de 0,79. En la ilustración 29, mostramos la matriz de confusión obtenida con dicho algoritmo con los datos de test, donde vemos que tenemos una mayor precisión al clasificar un autor como hombre:

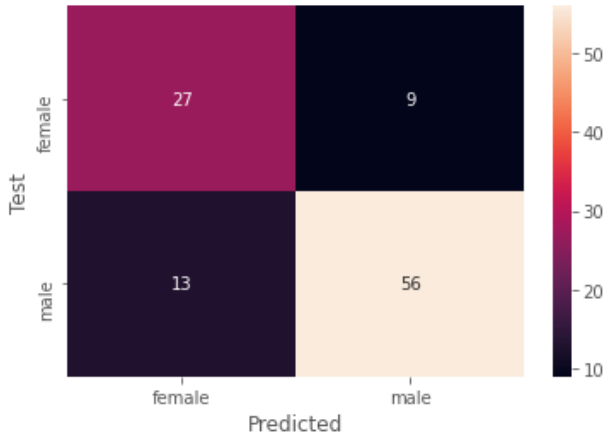

Ilustración 29. Matriz confusión LGBMClassifier empleando TDIDF (género)

6.1.3 Profesión

En la *tabla 22* se pueden ver los resultados obtenidos en la clasificación de la profesión de los autores de los tweets:

Modelo	Datos Entrada	f1_scores Test	Accuracy Test
LGBMClassifier	TDIDF	0,929293	0,952381
Red Neuronal	TDIDF	0,907760	0,923810
SVM	TDIDF	0,900285	0,923810
LogisticRegression	TDIDF	0,868750	0,904762
MLPClassifier	TDIDF	0,851190	0,904762
BERT	TWEET	0,838789	0,809523
RandomForest	TDIDF	0,765326	0,866667
Red Neuronal Word2Vec	TWEET_LEMA	0,764757	0,761905
MLPClassifier	METRICAS	0,657355	0,761905
Red Neuronal	METRICAS	0,571429	0,761905
LogisticRegression	METRICAS	0,432432	0,761905
RandomForest	METRICAS	0,432432	0,761905
SVM	METRICAS	0,432432	0,761905
BASELINE	BOW	0.411765	0.700000
LGBMClassifier	METRICAS	0,192308	0,238095

Tabla 22. Resultados clasificación por profesión, tweets agrupados por autor

(Se aplica una escala de color a los datos, más rojo cuanto el valor es más pequeño y más azul a valores más grandes. En gris los resultados de la baseline)

Para la profesión también obtenemos buenos resultados, en todos los casos excepto uno por encima del baseline, tanto LGBMClassifier, la red neuronal secuencial como SVM utilizando el vector TDIDF obtienen resultados por encima del 0,90 para todas las métricas. De todos ellos es LGBMClassifier el que mejor se comporta y si observamos su matriz de confusión (ilustración 30) vemos que

sólo tenemos falsos positivos cuando intentamos clasificar como políticos algunos periodistas.

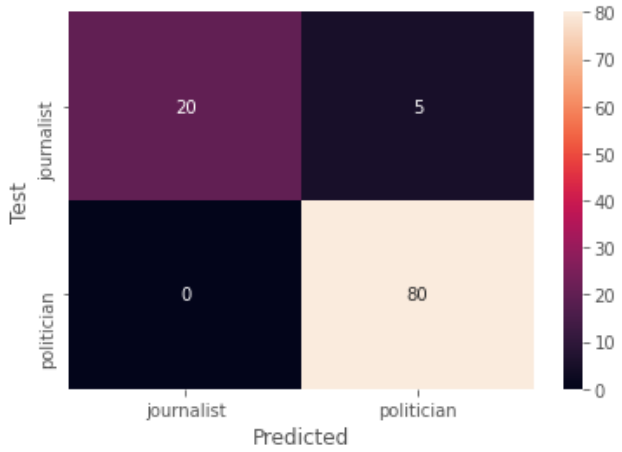

Ilustración 30. Matriz confusión LGBMClassifier empleando TDIDF (profesión)

Volvemos a obtener los peores resultados tomando como entrada los datos de métricas como datos de entrada y con BERT llegamos a alcanzar un f11-score de 0.83 comportándose mejor que en el resto de las clasificaciones.

6.1.4 Ideología Multiclase

A continuación, mostramos en la *tabla 23* los resultados obtenidos en la clasificación de la ideología multiclase:

Modelo	Datos Entrada	f1_scores Test	Accuracy Test
Red Neuronal	TDIDF	0,803425	0,790476
SVM	TDIDF	0,800749	0,790476
LogisticRegression	TDIDF	0,762876	0,761905
LGBMClassifier	TDIDF	0,747419	0,761905
MLPClassifier	TDIDF	0,733934	0,752381
RandomForest	TDIDF	0,732277	0,771429
BERT	TWEET	0,394009	0,371428
BASELINE	BOW	0.267857	0.450000
Red Neuronal Word2Vec	TWEET_LEMA	0,247272	0,276190
SVM	METRICAS	0,127660	0,342857
Red Neuronal	METRICAS	0,119318	0,200000
LGBMClassifier	METRICAS	0,111111	0,285714
MLPClassifier	METRICAS	0,109848	0,276190
RandomForest	METRICAS	0,083333	0,200000
LogisticRegression	METRICAS	0,069672	0,161905

Tabla 23. Resultados clasificación ideología multiclase, tweets agrupados por autor

(Se aplica una escala de color a los datos, más rojo cuanto el valor es más pequeño y más azul a valores más grandes. En gris los resultados de la baseline)

A pesar de ser una clasificación multilabel obtenemos unos buenos resultados, ligeramente peores que para la ideología binaria, como era de esperar, pero bastante por encima del baseline. La red neuronal y SVM empleando el vector TDIDF obtienen valores similares siendo ligeramente mejores en la red neuronal con la que alcanzamos un f1-score de 0,80 con los datos de test. Si observamos su matriz de confusión (ilustración 31) vemos que donde tenemos mayores dificultades es en diferenciar entre las ideologías moderadas de sus correspondientes ideologías "más extremistas":

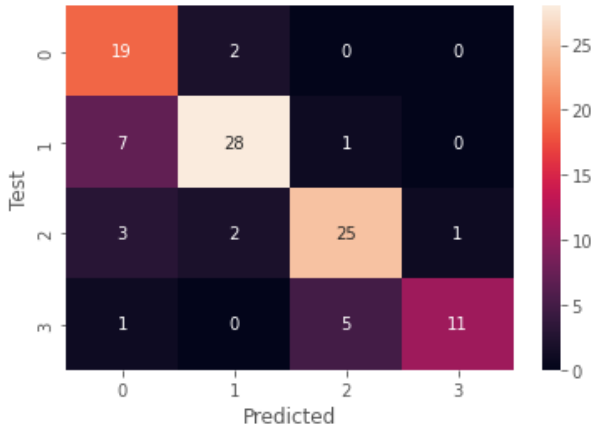

Ilustración 31. Matriz confusión red neuronal empleando TDIDF, ideología multiclase

En este caso, los resultados obtenidos cuando tomamos como entrada las métricas son bastante malos, peores incluso que en los casos anteriores, al igual que cuando empleamos BERT y Word2Vec donde ni el f1-score ni la accuracy llegan a alcanzar 0,40.

6.2 Resultados obtenidos a nivel de tweet

Además de la clasificación a nivel de autor se ha realizado una clasificación extra a nivel de tweet individual. La evaluación la realizamos a partir del dataset de test aplicando a cada tweet la clasificación del autor correspondiente. En este caso los resultados obtenidos han sido a nivel general peores que en el caso anterior:

6.2.1 Ideología Binaria

En este apartado se evaluarán los resultados obtenidos en la clasificación de ideología binaria, los cuales se pueden ver en la *tabla 24*.

Modelo	Datos Entrada	f1_scores Test	Accuracy Test
BERT	TWEET	0,627568	0,627857
SVM	TDIDF	0,606974	0,616349
LogisticRegression	TDIDF	0,605262	0,615317
LGBMClassifier	TDIDF	0,600438	0,609286
Red Neuronal	TDIDF	0,595844	0,596746
RandomForest	TDIDF	0,593605	0,616905
MLPClassifier	TDIDF	0,571911	0,605397
Red Neuronal Word2Vec	TWEET_LEMA	0,538004	0,517937
LogisticRegression	METRICAS	0,529457	0,531032
Red Neuronal	METRICAS	0,528494	0,528571
LGBMClassifier	METRICAS	0,528075	0,530238
RandomForest	METRICAS	0,525523	0,529841
SVM	METRICAS	0,524257	0,527778
MLPClassifier	METRICAS	0,504491	0,521508

Tabla 24. Resultados clasificación ideología binaría, tweets individuales
(se aplica una escala de color a los datos, más rojo cuanto el valor es más pequeño y más azul a valores más grandes)

Los mejores resultados los obtenemos empleando BERT, la diferencia entre emplear las métricas o emplear TDIDF se reduce respecto cuando analizábamos los tweets agrupados por autor, pero sigue siendo mejor TDIDF. La red neuronal y MLPClassifier que habían sido los que mejor resultado habían obtenido con los tweets agrupados por autor, obtienen en este caso peores resultados comparados con los otros algoritmos.

La matriz de confusión (ilustración 32) obtenida con el método con el que hemos obtenido los mejores resultados, BERT, es la siguiente (la precisión es algo mejor al clasificar tweets de “izquierdas” 0,65 que de derechas 0,59):

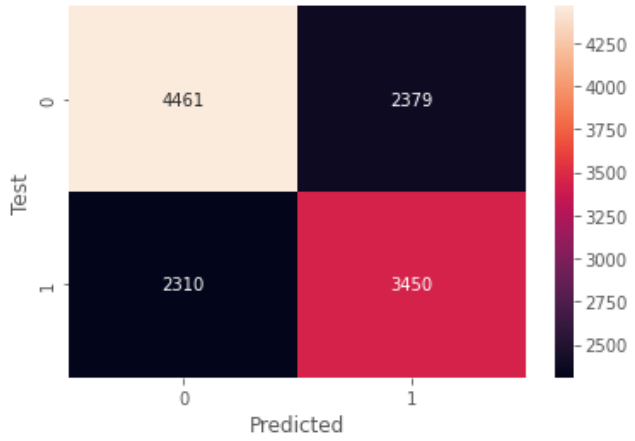

Ilustración 32. Matriz confusión empleando BERT, ideología binaria (0-left, 1-right)

6.2.2 Género

A continuación, mostramos los resultados, *tabla 25*, obtenidos en la clasificación del género de los autores.

Modelo	Datos Entrada	f1_scores Test	Accuracy Test
BERT	TWEET	0,631849	0,631667
SVM	TDIDF	0,559222	0,603889
RandomForest	TDIDF	0,559112	0,628095
LGBMClassifier	TDIDF	0,553799	0,607619
Red Neuronal	METRICAS	0,551351	0,550079
LogisticRegression	TDIDF	0,549654	0,588571
Red Neuronal Word2Vec	TWEET_LEMA	0,533418	0,533333
Red Neuronal	TDIDF	0,530916	0,530873
MLPClassifier	TDIDF	0,529451	0,575159
RandomForest	METRICAS	0,523824	0,557063
LGBMClassifier	METRICAS	0,523101	0,558175
LogisticRegression	METRICAS	0,519363	0,545159
SVM	METRICAS	0,519255	0,546508
MLPClassifier	METRICAS	0,510250	0,553492

Tabla 25. Resultados clasificación por género, tweets individuales

(se aplica una escala de color a los datos, más rojo cuanto el valor es más pequeño y más azul a valores más grandes)

En este caso nos encontramos en una situación parecida al caso anterior. BERT obtiene de nuevo las mejores métricas con un f1-sore y accuracy por encima de 0,6. El resultado es bastante parecido para todos los métodos empleados (diferencias entre el peor y mejor inferiores a 0.13 tanto en f1-score como en accuracy).

En la ilustración 33 mostramos la matriz de confusión obtenida con BERT, donde vemos que se comporta mejor al clasificar el género masculino, alcanzando un 0.68 de precisión:

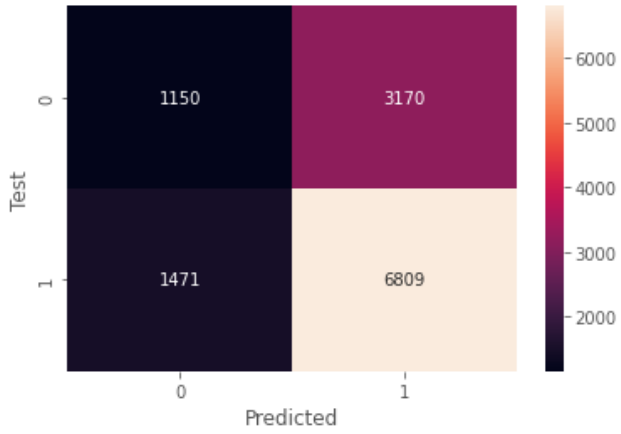

Ilustración 33. Matriz confusión empleando BERT, género (0-female,1-male)

6.2.3 Profesión

En la tabla *tabla 26* mostramos los resultados obtenidos en la clasificación del género a partir de cada tweet individual.

Modelo	Datos Entrada	f1_scores Test	Accuracy Test
BERT	TWEET	0,757431	0,757619
Red Neuronal Word2Vec	TWEET_LEMA	0,729616	0,729524
Red Neuronal	METRICAS	0,625524	0,625635
RandomForest	TDIDF	0,606706	0,663095
Red Neuronal	TDIDF	0,604664	0,604762
LGBMClassifier	TDIDF	0,596936	0,640556
SVM	TDIDF	0,593231	0,638492
RandomForest	METRICAS	0,591827	0,680476
LogisticRegression	TDIDF	0,591452	0,630794
LGBMClassifier	METRICAS	0,587165	0,656429
SVM	METRICAS	0,579367	0,697063
LogisticRegression	METRICAS	0,564713	0,615952
MLPClassifier	TDIDF	0,525929	0,727302
MLPClassifier	METRICAS	0,512564	0,736746

Tabla 26. Resultados clasificación por profesión, tweets individuales
(se aplica una escala de color a los datos, más rojo cuanto el valor es más pequeño y más azul a valores más grandes)

Es la clasificación donde mejores resultados obtenemos al analizar los tweets de forma individual, de nuevo BERT es el que mejor se comporta (una accuracy y f1-score por encima de 0,75). También destacar que en este caso si obtenemos unos buenos resultados empleando Word2Vec y utilizando las métricas de los textos como entrada de la red neuronal obtenemos resultados por encima de 0.6, siendo el tercer algoritmo que mejor se comporta atendiendo al f1-score de test.

En la ilustración 34 mostramos la matriz de confusión obtenida con BERT, siendo bastante baja la precisión cuando clasificamos periodistas, solo un 0.48, mientras que para politicos llega al 0.80:

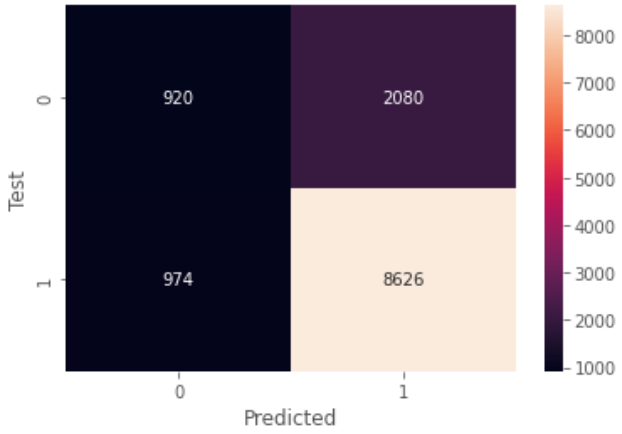

Ilustración 34. Matriz confusión empleando BERT,profesión (0-journalist, 1-politician)

6.2.4 Ideología Multiclase

En este apartado evaluamos los resultados de la clasificación de ideología política multiclase de cada tweet individual, mostrados en la tabla 27.

Modelo	Datos Entrada	f1_scores Test	Accuracy Test
BERT	TWEET	0,406551	0,406984
LogisticRegression	TDIDF	0,357170	0,375556
SVM	TDIDF	0,349706	0,359524
RandomForest	TDIDF	0,345778	0,361032
LGBMClassifier	TDIDF	0,338154	0,340317
MLPClassifier	TDIDF	0,324899	0,377460
RandomForest	METRICAS	0,273000	0,290397
LGBMClassifier	METRICAS	0,271782	0,286270
SVM	METRICAS	0,267748	0,279683
MLPClassifier	METRICAS	0,252916	0,300238
LogisticRegression	METRICAS	0,252693	0,305794
Red Neuronal	TDIDF	0,197164	0,340159
Red Neuronal Word2Vec	TWEET_LEMA	0,059435	0,298810
Red Neuronal	METRICAS	0,002390	0,274206

Tabla 27. Resultados clasificación ideología multiclase, tweets individuales
(se aplica una escala de color a los datos, más rojo cuanto el valor es más pequeño y más azul a valores más grandes)

Para la clasificación ideológica multiclase obtenemos los peores resultados. La accuracy y el f1-score están lejos de alcanzar el 0,50 (nos quedamos en 0,40). BERT es de nuevo el que mejor se comporta, pero en general no logramos buenos resultados.

La matriz de confusión obtenida con BERT se muestra en la ilustración 35, donde se observa la dificultad en distinguir entre perfiles moderados (moderado de izquierdas

vs moderado de derechas) y entre los perfiles moderados y sus correspondientes ideologías más extremistas:

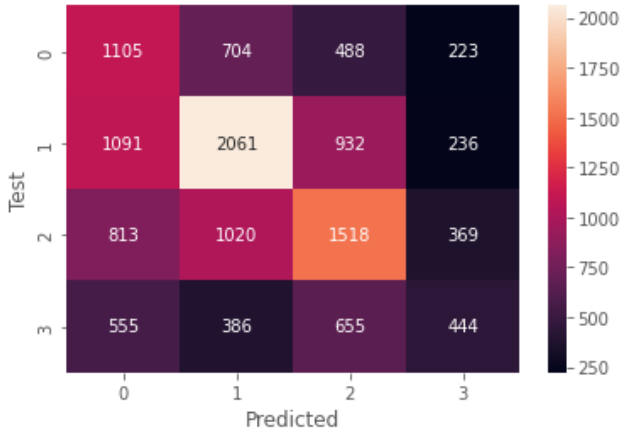

Ilustración 35. Matriz confusión empleando BERT, ideología multiclase

6.3 Resumen resultados

A modo resumen mostramos en las siguientes gráficas los resultados de las métricas f1-score obtenidos por cada método con el dataset de test:

- Tweets agrupados por autor (ilustración 36): De media empleando redes neuronales, LGMClassifier y SVM obtenemos los mejores resultados. También observamos los malos resultados obtenidos cuando hemos empleado las métricas de los textos para la clasificación y los discretos resultados empleando BERT y Word2Vec.

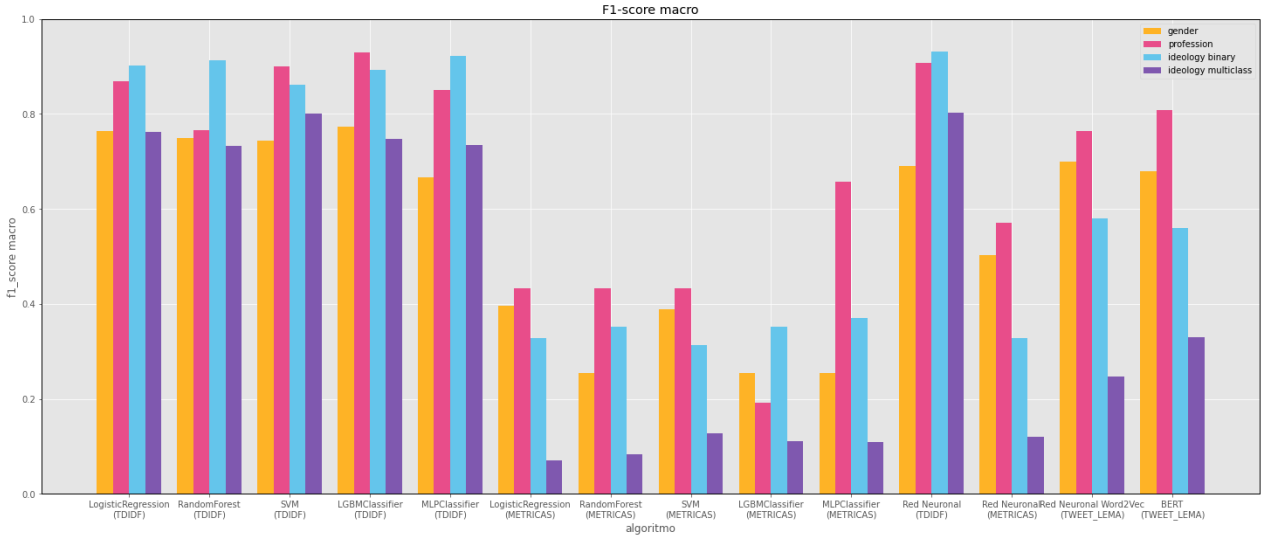

Ilustración 36. Gráfica con el f1-score macro obtenido por cada algoritmo y para clasificación realizada sobre los tweets de cada autor

- Tweets individuales (ilustración 37): donde BERT obtiene los mejores resultados en todos los casos y a nivel general los resultados son bastantes más discretos que en el caso anterior (tener en cuenta que el nombre de los partidos políticos y de las cuentas de Twitter están anonimizados). Solo llegamos a un f1-score superior a 0,7 al clasificar por la profesión empleando BERT y Word2Vec.

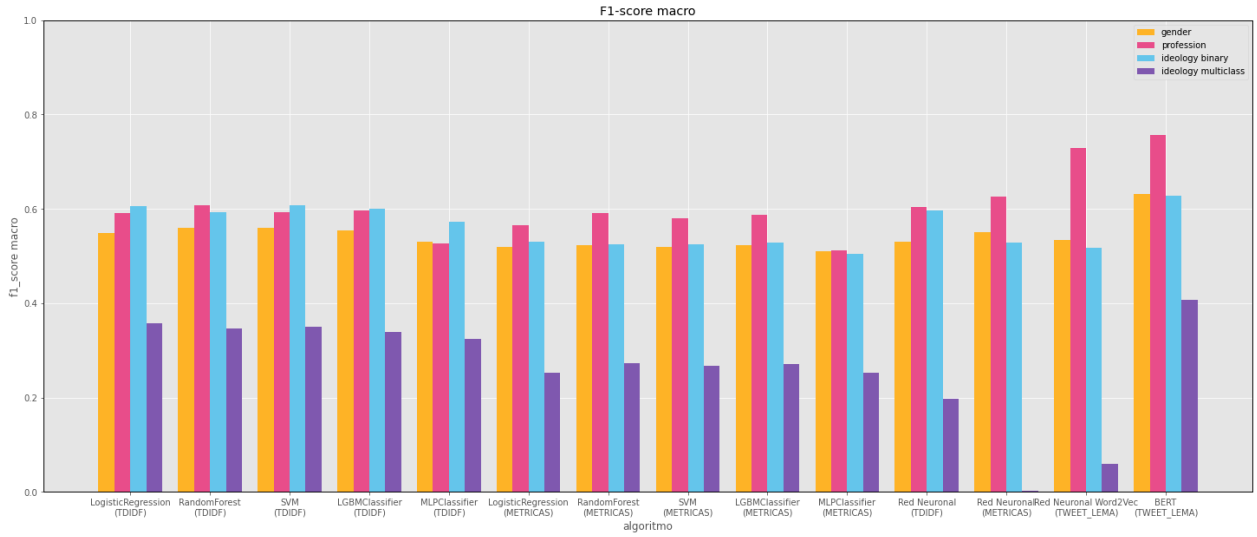

Ilustración 37. Gráfica con el f1-score macro obtenido por cada algoritmo y para clasificación realizada a cada tweet individual

6.4 Comparativa con los resultados obtenidos en la competición IberLEF

A continuación, compararemos los resultados obtenidos con los del resto de trabajos presentados en la competición.

En la *tabla 28* se muestran los 10 mejores resultados obtenidos en la fase de evaluación de la competición (hay más de 20 trabajos remitidos), junto con nuestros resultados, donde observamos que nuestro trabajo estaría en el top 5 de mejores resultados en la fase de "evaluación" de la tarea CodaLab:

	User	Average F1	F1 Gender	F1 Profession	F1 Ideology Binary	F1 Ideology Multiclass
1	sesantam	0.902262 (1)	0.902868 (1)	0.944327 (1)	0.961623 (1)	0.800229 (5)
2	villacu	0.890961 (2)	0.784836 (6)	0.921250 (4)	0.961482 (2)	0.896275 (1)
3	amsqr	0.889182 (3)	0.826714 (3)	0.933452 (2)	0.951519 (3)	0.845044 (3)
4	enriquesec	0.879757 (4)	0.836830 (2)	0.895000 (6)	0.941667 (4)	0.845530 (2)
5	juancarperez	0,859464 (5)	0,773173 (9)	0,929292 (3)	0,931967 (5)	0,803424 (4)
6	CristinaGH	0.825320 (6)	0.726020 (14)	0.897760 (5)	0.921759 (6)	0.755739 (7)
7	Bernardo	0.819963 (7)	0.791775 (4)	0.849818 (9)	0.913151 (7)	0.725108 (9)
8	pablopcr	0.799984 (8)	0.743767 (12)	0.867561 (8)	0.862153 (10)	0.726455 (8)
9	CYAM	0.798488 (9)	0.782222 (7)	0.826811 (12)	0.821429 (12)	0.763492 (6)
10	andrei.manea	0.786940 (10)	0.737976 (11)	0.883463 (7)	0.902198 (8)	0.624124 (13)
11	joseluisUS	0.781640 (11)	0.791775 (4)	0.795322 (14)	0.913151 (7)	0.626312 (12)

Tabla 28. Mejores resultados obtenidos en la fase de evaluación de la competición, entre paréntesis el puesto de la clasificación según la columna correspondiente, junto con nuestros resultados. (fuente: https://codalab.lisn.upsaclay.fr/competitions/1948#results)

Señalar que también que hemos superado con creces los resultados obtenidos en el baseline suministrado en la competición como se observa en la tabla 29.

User	Average F1	F1 Gender	F1 Profession	F1 Ideology Binary	F1 Ideology Multiclass
juancarperez	0,859464 (5)	0,773173 (9)	0,929292 (3)	0,931967 (5)	0,803424 (4)
BASELINE	0,423790	0,466667	0,411765	0,548872	0,267857

Tabla 29. Resultados obtenidos en la fase de "evaluación" en nuestro trabajo, entre paréntesis el puesto que hubiésemos obtenido según la columna correspondiente, junto con los resultados del baseline

A continuación, se hizo una afinación de grano fino de los hiperparámetros de los algoritmos con mejor f1-score sobre los datos de entrenamiento (LGBMClassifier para el género y la profesión y la red neuronal para la ideología política tanto binaria como multiclase), en la *tabla 30*, se muestra el resultado final obtenido:

label	f1_score	best
gender	0,782222	LGBMClassifier(is_unbalance=True, learning_rate=0.4, max_depth=5, n_estimators=150)
profession	0,931471	LGBMClassifier(is_unbalance=True, learning_rate=0.4, max_depth=6, n_estimators=150)
ideology_binary	0,941667	{'embeddings': False, 'epochs': 125, 'learn_rate':0.000135,'n_features':5000,'numClasses':2}
ideology_multiclass	0,821585	{'embeddings': False, 'epochs': 120, 'learn_rate':0.0051,'n_features':5000,'numClasses':4}

Tabla 30. Resultados finales obtenidos para cada tipo de clasificación.

Finalmente se subieron los resultados anteriores a la fase de “Post-Evaluación” de la competición (usuario juancarperez), con ello se logró alcanzar el primer lugar de dicha fase (tabla 31), con un f1-soce medio de 0.869236, en el momento de redacción de la presente memoria teniendo en cuenta la media de la métrica f1-score macro obtenida para cada clase:

Results									
#	User	Entries	Date of Last Entry	Team Name	Average Macro F1 ▲	F1 Gender ▲	F1 Profession ▲	F1 Ideology Binary ▲	F1 Ideology Multiclass ▲
1	juancarperez	5	08/15/22	TFM	0.869236 (1)	0.782222 (2)	0.931471 (1)	0.941667 (2)	0.821585 (2)
2	hiramcp	13	05/10/22	NFOTEC-LaBD	0.842911 (2)	0.746377 (3)	0.833309 (3)	0.961312 (1)	0.830646 (1)
3	CYAM	3	05/09/22	TeamMX	0.823807 (3)	0.791775 (1)	0.826811 (4)	0.913151 (3)	0.763492 (3)
4	andrei.manea	2	05/28/22		0.777186 (4)	0.737976 (4)	0.844444 (2)	0.902198 (4)	0.624124 (4)
5	JoseAGD	1	05/08/22	UMUTeam	0.511228 (5)	0.576211 (5)	0.432432 (6)	0.595665 (6)	0.440603 (5)
6	DarkvidMP	1	06/10/22		0.488200 (6)	0.539978 (6)	0.579920 (5)	0.668389 (5)	0.164512 (6)

Tabla 31. Resultados fase post-evaluación de la competición (nuestro trabajo “juancarperez”). Entre paréntesis, la clasificación de los resultados según la métrica indicada por la columna (fuente: https://codalab.lisn.upsaclay.fr/competitions/1948#results)

7. CONCLUSIONES Y TRABAJOS FUTUROS

En el siguiente apartado expondremos nuestras conclusiones, así como trabajos futuros con los que mejorar y completar el presente trabajo.

7.1 Conclusiones

En el presente TFM hemos puesto en práctica los conocimientos adquiridos a lo largo del Master de Ingeniería y Ciencia de Datos. Dicho trabajo nos ha permitido explorar las capacidades para la clasificación de textos de forma automática, lo cual podremos extrapolar al mundo laboral donde se dispone de gran cantidad de información en textos (legales, contractuales, partes de siniestros, …) que es necesario clasificar, analizar y transformar en información estructurada para poder explotar dicha información.

A partir de los datasets publicados en la competición "IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology" se ha realizado la extracción del género, la profesión y la ideología política, binaria y multiclase, de los autores. Para ello se han ido completando una serie de tareas con el fin de analizar y normalizar los textos de entrada y entrenar, evaluar y afinar distintos modelos de clasificación estudiados durante el presente Master. Con ello finalmente, empleando una red neuronal y LGBMClassifier, se ha conseguido alcanzar los primeros puestos en la competición en la fase de post-evaluación para cada una de las clasificaciones (alcanzamos el primer puesto si tenemos en cuenta la media de las métricas obtenidas en cada clasificación). Se ha logrado alcanzar una f1-score media por encima del 0,869236 (tabla 32) y los resultados obtenidos están bastante por encima del baseline de la competición que presenta un f1-score de 0,423790.

Results									
#	User	Entries	Date of Last Entry	Team Name	Average Macro F1 ▲	F1 Gender ▲	F1 Profession ▲	F1 Ideology Binary ▲	F1 Ideology Multiclass ▲
1	juancarperez	5	08/15/22	TFM	0.869236 (1)	0.782222 (2)	0.931471 (1)	0.941667 (2)	0.821585 (2)

Tabla 32. Resultados de nuestro trabajo en la fase post-evaluación de la competición "IberLEF 2022 Task PoliticEs. Spanish Author Profiling for Political Ideology"

También se ha realizado una clasificación teniendo en cuenta cada tweet de forma individual, en donde los resultados han sido más discretos. BERT es el algoritmo que mejores resultados ha logrado en todos los casos, alcanzado un f1-score de 0.75 para la profesión, sobre el 0.63 para el género y la ideología política binaria mientras que en clasificación multiclase sólo hemos conseguido alcanzar un 0.4, donde hemos visto que le cuesta diferenciar entre las ideologías políticas más moderadas de las "más" extremistas.

A nivel personal destacaría el hecho de subir los resultados a una competición de IberLEF/Codalab, algo que ni me imaginaba antes de empezar a cursar este Master. Además, señalaría la dificultad de encontrar modelos BERT preentrenados y vectores Word2Vec y librerías para procesar textos en español, habiendo múltiples de opciones para el inglés y no tantas para el castellano.

7.2 Trabajos Futuros

Al presente trabajo se le pueden incorporar una variedad de mejoras para que nuestro sistema tenga un mejor rendimiento y/o podamos obtener unos resultados más amplios, por ejemplo:

- En cuanto a la clasificación en sí, mejorar los resultados obtenidos mediante el estudio de otros métodos de clasificación. Por ejemplo, para mejorar los resultados de BERT en la clasificación de tweets agrupados al estar limitado a 512 tokens podríamos:
  - truncar por la mitad del texto en lugar de por el principio
  - probar la solución que hemos visto en otros trabajos en el apartado 2 donde los tweets se agrupan en bloques de 512 tokens, cada bloque se clasifica por separado y empleando un sistema de votación se obtiene la clasificación final
  - utilizar otros Transformers como Longformer o Reformer que no presentan esta limitación.

- Desarrollar una aplicación web donde explotar los modelos desarrollados, desde la cual se puedan subir los tweets de un usuario y determinar la clasificación obtenida (género, profesión, ideología política). Incluso podríamos ir un paso más allá y se pudiese subir un dataset etiquetado y el sistema devuelva los resultados obtenidos y el algoritmo que mejor se ha comportado en base a una métrica indicada.

- Podríamos ampliar el sistema para tomar como los datos de entrada en otro idioma a parte del castellano y observar cómo se comporta.

- Se podría determinar los usuarios autores de los tweets de los datasets de entrada y recuperar los tweets originales. De este modo podríamos completar el estudio usando técnicas descritas en la asignatura de Minería de Datos en las Redes Sociales, empleando para ello por ejemplo Gephi.

BIBLIOGRAFÍA

Para este proyecto se han consultado a parte de los apuntes del curso las siguientes fuentes:

[1]. *García-Díaz, J. A., Colomo-Palacios, R., & Valencia-García, R. (2022). Psychographic traits identification based on political ideology: An author analysis study on Spanish politicians' tweets posted in 2020. Future Generation Computer Systems, 130(1), 59-74.*

[2]. *Aurélien Géron (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition:* https://learning.oreilly.com/library/view/hands-on-machine-learning/9781492032632

[3]. *Alice Zheng, Amanda Casari (2018). Feature Engineering for Machine Learning:* https://learning.oreilly.com/library/view/feature-engineering-for/9781491953235

[4]. *Hannes Hapke, Hobson Lane, Cole Howard (2019). Natural Language Processing in Action:* https://learning.oreilly.com/library/view/natural-language-processing/9781617294631

[5]. *Wes McKinney (2017). Python for Data Analysis, 2nd Edition:* https://learning.oreilly.com/library/view/python-for-data/9781491957653/

[6]. *V.Ñ. Vapnik. New York: Springer-Verlag (1995). The nature of statistical learning theory.* https://www.vebuso.com/2020/03/svm-hyperparameter-tuning-using-gridsearchcv

[7]. *Tomas Mikolov (2018).Efficient Estimation of Word Representations in Vector Space*

[8]. *Tomas Mikolov (2013). Distributed representations of words and phrases and their compositionality*

[9]. *Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding:* https://arxiv.org/pdf/1810.04805.pdf

[10]. *Fatke, M. (2017). Rasgos de personalidad e ideología política: una primera valoración global.* https://onlinelibrary.wiley.com/doi/abs/10.1111/pops.12347

[11]. *Abooraig R., Al-Zu'bi S., Kanan T., Hawashin B., Al Ayoub M., Hmeidi I."Automatic categorization of Arabic articles based on their political orientation"*

[12]. *Dahllöf M. Automatic prediction of gender, political affiliation, and age in Swedish politicians from the wording of their speeches—A comparative study of classifiability*

[13]. *Koppel M., Schler J., Argamon S. Computational methods in authorship attribution*

[14]. *Ramy Baly, Giovanni Da San Martino, James Glass, Preslav Nakov. We can detect your bias: Predicting the political ideology of news articles.*

[15]. *Sergio Santamaria Carrasco, Roberto Cuervo Rosillo. Political Author Profiling using BETO and MarIA*

[16]. Emilio Villa Cueva, Ivan González Franco, Fernando Sanchez Vega and Adrián Pastor López Monroy. NLP-CIMAT at PoliticEs 2022: PolitiBETO, a Domain- Adapted Transformer for Multi-class Political

[17]. Alejandro Mosquera. Alejandro Mosquera at PoliticEs 2022: Towards Robust Spanish Author Profiling and Lessons Learned from Adversarial Attacks

[18]. Enrique Santibáñez Cortés, Azael Carrillo Cabrera, Yair Antonio Castillo Castillo, Daniela Moctezuma and Victor Muñiz-Sánchez. Ensemble Based Classification Algorithms for Author Profiling in Spanish Language

REFERENCIAS WEB

[HTTP1]. https://codalab.lisn.upsaclay.fr/competitions/1948#learn_the_details

[HTTP2]. https://arxiv.org/abs/2109.04870

[HTTP3]. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[HTTP4]. https://www.ibm.com/cloud/learn/random-forest

[HTTP5].https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[HTTP6]. https://scikit-learn.org/stable/modules/svm.html

[HTTP7]. https://scikit-learn.org/stable/modules/neural_networks_supervised.html

[HTTP8]. https://lightgbm.readthedocs.io/

[HTTP9]. https://lightgbm.readthedocs.io/en/latest/Parameters-Tuning.html

[HTTP10].https://www.analyticsvidhya.com/blog/2021/05/tuning-the-hyperparameters-and-layers-of-neural-network-deep-learning/

[HTTP11]. https://github.com/google-research/bert

[HTTP12]. https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec

[HTTP13].https://medium.com/qu4nt/reducir-el-n%C3%BAmero-de-palabras-de-un-texto-lematizaci%C3%B3n-y-radicalizaci%C3%B3n-stemming-con-python-965bfd0c69fa

[HTTP14]. https://github.com/dccuchile/beto

[HTTP15]. https://github.com/google-research/bert/blob/master/multilingual.md

[HTTP15]. *https://www.vebuso.com/2020/03/svm-hyperparameter-tuning-using-gridsearchcv/*

ANEXO

Código Fuente

El código fuente de este TFM se encuentra disponible en el siguiente enlace:

https://colab.research.google.com/drive/16DVaj6vWDv-aawBeL9JrGtpXUo2QT56D?usp=sharing

El código está comentado y dividido en secciones, las cuales corresponden a cada una de fases en las que hemos dividido el proyecto:

Ilustración 38. Estructura código fuente del TFM

Software Empleado

El presente trabajo ha sido desarrollado empleando Python, lenguaje de programación creado por Guido van Rossum a principios de los años 90 y consta de una licencia BSD que permite su uso de manera gratuita (https://www.python.org). Se ha empleado un cuaderno Jupyter para el desarrollo de todo el proyecto.

Las librerías principales que se han utilizado en la realización de este trabajo han sido:

- Pandas para la gestión de los datasets de entrada y salida
- Tensorflow y Keras para la implementación y ejecución de las redes neuronales
- Gensim para utilizar Word2Vec

- Transformers para poder emplear BERT
- Scikit-learn nos facilita la ejecución de las tareas de aprendizaje automático (algoritmos de clasificación, escalado y codificación de los datos de entrada, cálculo de métricas,….)
- NLTK para las tareas de procesamiento del lenguaje natural
- Stopwordsiso para la eliminación de las stopwords de los tweets
- Stanza para lemmatizar los tweets
- WordCloud para obtener las nubes de palabras más repetidas por cada clase
- Matplotlib para las gráficas

La instalación y ejecución de las librerías MultiAzterTest y TextComplexityFreeling con las que se obtienen las métricas de los textos se ha realizado en un equipo local con S.O. Ubuntu en el cual se ha instalado Anaconda, mientras que el resto del trabajo se ha realizado empleando soluciones de la nube:

- Google Colab, desde donde lanzamos el cuaderno Jupyter con el código del presente trabajo (también se realizaron pruebas con PaperSpace, https://console.paperspace.com, pero finalmente se optó por emplear Google Colab principalmente por estar más familiarizado con su uso)

- Google Drive donde almacenamos los datos de entrada y los de salida

Datasets entrada

En la siguiente ruta se encuentran los dataset de entrada obtenidos de la competición (https://codalab.lisn.upsaclay.fr/competitions/1948#participate-get_starting_kit) así como los dataset con las transformaciones realizadas en la fase de normalización (lematizacion, eliminación de stopwords,…) y el cálculo de las métricas de los tweets (empleando las librerías MultiAzterTest y TextComplexityFreeling):

- https://drive.google.com/drive/folders/1WNXKyzF50QAZhUVuzsbQcpZU76s4u4qm?usp=sharing

Datos generados

En el siguiente enlace se pueden encontrar los ficheros csv con el resultado de las ejecuciones y los f1-score, accuracy y tiempos de ejecución de cada algoritmo:

- https://drive.google.com/file/d/1izJ1IMInjlj6s4YbJBSZsCLs0kW7ol8O/view?usp=sharing

Estudio accuracy en virtud de la longitud de los tweets

Al observar los resultados no muy buenos cuando clasificamos cada tweet individual se realizó un estudio para ver la influencia de la longitud del tweet y su correcta clasificación. Dicho estudio se realizó a partir de la clasificación realizada por BERT, dado a que ha sido el que mejores resultados ha obtenido a nivel de tweet. En la

ilustración 39 se puede observar que la “accuracy” tiende a mejorar con el tamaño del tweet a clasificar:

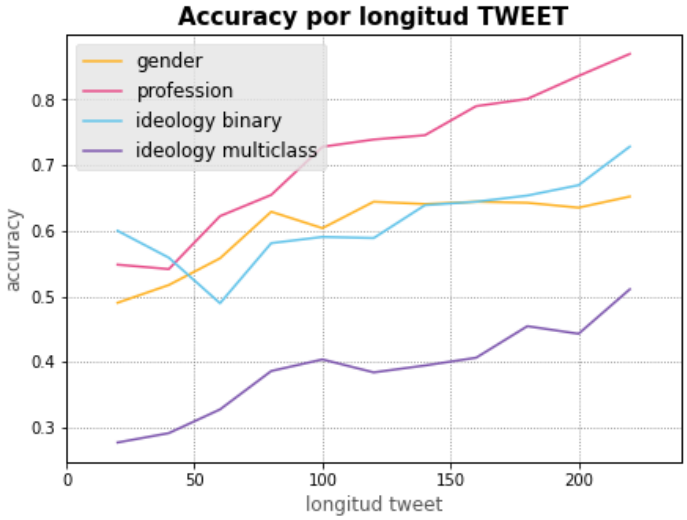

Ilustración 39. Gráfica relación accuracy y longitud de los tweets