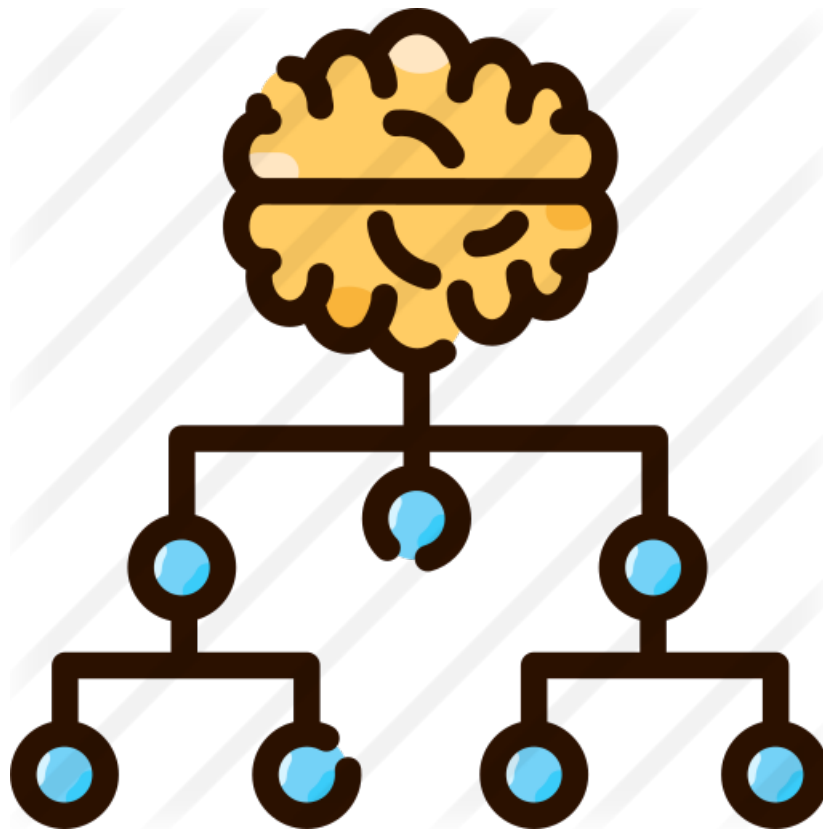


Práctica 2: Clasificador Documental

Ingeniería Lingüística

Máster Universitario en Inteligencia Artificial 2020/21



Ana M^a Casado Faulí
Ana Sanmartín Domenech

Índice

1. Introducción	3
1.1. Descripción del problema	3
1.2. Estructura de la práctica	3
2. Obtención de los documentos	3
2.1. Método de búsqueda de información	4
2.2. Documentos almacenados	4
3. Extracción del glosario representativo	7
3.1. Selección de los términos relevantes	8
4. Clasificador de documentos	9
4.1. <i>VSM</i>	9
4.2. <i>TF-IDF</i>	9
4.3. <i>Naïve Bayes</i>	10
4.4. <i>Similaridad del coseno</i>	11
5. Implementación	11
5.1. Instalación para <i>Windows</i>	12
5.2. Instalación para <i>Ubuntu (Linux)</i>	12
5.3. Guía de Usuario	12
5.3.1. Estructura del proyecto	12
5.3.2. Dependencias	13
5.4. Flujo de trabajo (<i>Workflow</i>) del proyecto	14
6. Resultados y Evaluación de los métodos	16
6.1. Cálculo de la bondad (Precisión, Exactitud, Sensibilidad/Exhaustividad y valor F1)	16
6.2. Clasificación con el modelo Doc2Bow y la métrica TFIDF y similaridad del coseno	17
6.3. Clasificación con el modelo Word2Vec y la métrica de la similaridad del coseno	19
6.4. Clasificación con el modelo CounterVectorizer y la métrica de Naïve Bayes	21
6.5. Comparativa	23
7. Conclusiones	23
Bibliografía	24
A. Listado de los documentos utilizados	25
B. Glosario de términos por tema	38
C. Selección a mano del glosario	39
D. Stop-words de <i>spacy</i>	41

1. Introducción

1.1. Descripción del problema

En la presente práctica se desea **diseñar e implementar un clasificador documental con glosario**. Para poder crear exitosamente un glosario representativo de un tema se necesitará obtener un gran conjunto de documentos representativos de dicho tema. Los temas deben de ser suficientemente disjuntos para ser fácilmente clasificables.

Un glosario debe contener los términos representativos de una temática, para ello no debe estar formado por *conceptos* sino por términos (palabras). Un glosario también debe incluir la *variabilidad lingüística* de los términos de referencia, la cual se hace mediante la variación morfológica.

1.2. Estructura de la práctica

La práctica se ha estructurado de la siguiente manera:

- ♥ En la sección 2 se explicará principalmente cómo se han obtenido los documentos.
- ♥ En la sección 3 se presentará el método utilizado para extraer las palabras para la creación del glosario representativo.
- ♥ En la sección 4 se presentará el clasificador y los métodos utilizados para obtener alguna métrica indicativa que identifique el dominio de cada documento.
- ♥ En la sección 5 se desarrollará paso a paso una guía de usuario.
- ♥ En la sección 6 se realizará una evaluación de los métodos.
- ♥ Finalmente, en la sección 7 se presentarán las conclusiones.

2. Obtención de los documentos

En esta sección se explicará brevemente las condiciones escogidas para filtrar la información, el método utilizado para buscarlos y los documentos elegidos.

Para poder extraer un glosario, los temas deben de ser bastante disjuntos, para ello, se ha elegido **deportes, salud y política**. En el proceso de la obtención de documentos se han tomado en cuenta las siguientes restricciones:

- ♥ **Tamaño del documento.** Los documentos a escoger deberán tener un mínimo contenido: **entre 400 y 1000 palabras**.
- ♥ **Origen (la fuente) del documento.** Para filtrar qué documentos serán aptos para su uso, se ha tomado la decisión de **aceptar solamente información proveniente de periódicos, revistas, o boletines**. Fuentes no oficiales, como blogs personales no se han considerado relevantes para ser parte del glosario.
- ♥ **Antigüedad del documento.** Puesto que en el mismo tema puede variar altamente en contenidos dependiendo de las noticias de actualidad y teniendo en cuenta el pequeño tamaño del glosario y documentos a generar, se ha considerado preferible tomar textos recientes. **El filtro se ha aplicado para los años 2019 y 2020**, con la intención de maximizar los resultados.

Para cada tema, se han extraído 30 documentos, dando la **totalidad de 90 documentos** ($30 \times 3 = 90$). Se ha elegido la primera mitad (documentos 1 al 15) de cada tema para generar el glosario y la segunda mitad (documentos 16 al 30) de cada tema para probar el modelo clasificador creado.

2.1. Método de búsqueda de información

Para realizar la búsqueda de los documentos, se ha optado por un método tradicional. Como puede observarse en la Figura 1, se ha realizado introduciendo la palabra clave (deportes, salud o política) en la sección de *Noticias* disponible en el *buscador Google*.

Se ha extraído el título y cuerpo de 30 noticias diferentes para cada tema y se ha almacenado el contenido en archivos de texto (.txt). La sintaxis de los documentos guardados sigue la estructura de: *temaN*, por ejemplo para el cuarto documento correspondiente a *Deportes* sería *deportes4*.

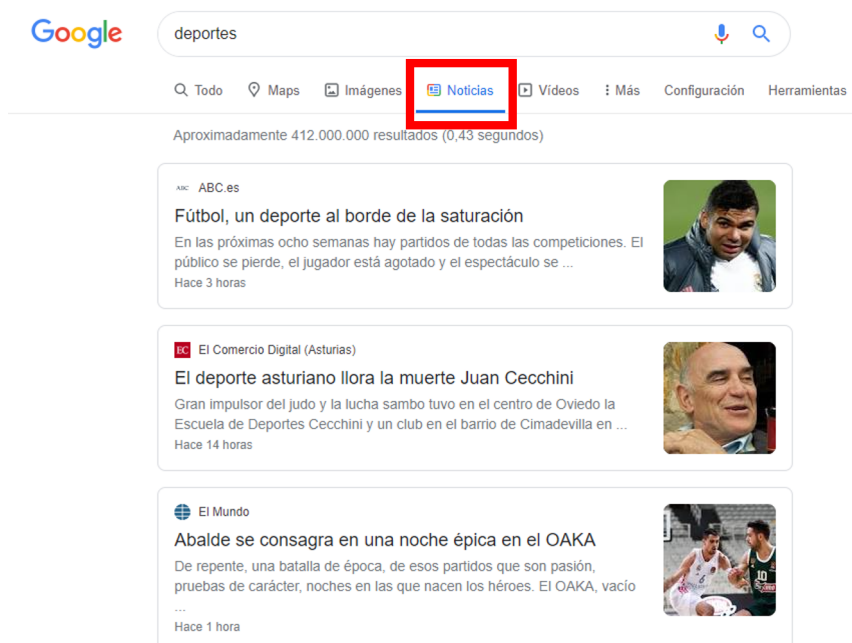


Figura 1: Forma de búsqueda de información utilizada en *Google Search News*

2.2. Documentos almacenados

A continuación se listarán los titulares de los artículos periodísticos escogidos para la creación del glosario (artículos 1 - 15) y para el test de clasificación de los mismos mediante el glosario (artículos 16 - 30). Así mismo, en el apéndice A se listará información más detallada sobre los documentos seleccionados (la fuente de donde se ha extraído la información, el autor, la fecha de publicación y la url dónde se puede acceder).

TÍTULAR DE LOS ARTÍCULOS PARA EL GLOSARIO Y TEST DEL TEMA *DEPORTES*.

1. El jugador de rugby de 42 años que no recuerda haber ganado el Mundial.
2. Barcelona - PSG y Atalanta - Real Madrid y Atleti - Chelsea, duelos de octavos de final en la Champions.
3. Gervasio Deferr: "El problema es que los que mandan no han hecho deporte en su vida".
4. El viejo truco que funciona otra vez a Zidane.
5. Messi sufre por el Barça.
6. El deporte federado vuelve a las canchas.
7. El deporte asturiano reclama al Gobierno regional que rectifique.
8. El Atalanta, una máquina de fútbol.

9. El deporte ya genera el 3,3 por ciento del PIB de España.
10. Alerta en el PSG por la nueva lesión del Neymar.
11. Coe defiende las protestas en los Juegos: "El deporte debe ser un catalizador".
12. Messi, el taconazo de Law y el derbi geoestratégico.
13. El Barça se descose.
14. La pandemia alumbra un atletismo de escaparate y récords.
15. El deporte en Sevilla: una buena práctica con mucho recorrido.
16. Andalucía recibe la bandera que la distingue como Región Europea del Deporte para 2021, que ve "un anhelo realizado".
17. El dictado de Kroos.
18. Alonso lidera el «rookie test» con un atracón de vueltas.
19. CP Miralvalle: un colegio con mucha clase.
20. Fútbol, un deporte al borde de la saturación.
21. Los deportes de invierno piden al Gobierno que las estaciones de esquí abran en Navidad.
22. No hay quien pare a Bada Huesca que se codea con los mejores en la ASOBAL.
23. El deporte español también se escribe en femenino.
24. La Copa del Rey de baloncesto 2021 se jugará en el Palacio de los Deportes de Madrid.
25. Muere Vince Reffet, el hombre volador, tras un accidente en Dubai.
26. Logroño Deporte aportará 250.000 euros para apoyar a los equipos de fútbol referentes.
27. Los deportes más antiguos de la historia.
28. Coronavirus en el deporte: El CSD no logra un acuerdo para un protocolo común de vuelta del deporte no profesional.
29. El deporte catalán denuncia a la Generalitat.
30. Sólo la segunda ola pudo derrotar a la promesa de la natación de Navia.

TÍTULAR DE LOS ARTÍCULOS PARA EL GLOSARIO Y TEST DEL TEMA SALUD.

1. ¿Qué es la esclerosis lateral amiotrófica (ELA)?: Síntomas, causas y cómo tratas la enfermedad.
2. Calendarios vacunales: así son los de Portugal, Italia, Reino Unido y España.
3. Cataluña registra el mayor aumento de plazas de FSE respecto a la última convocatoria.
4. "Objetivos cumplidos" en CAR-T dos años después de su integración en España.
5. La COVID-19 persistente ya tiene un código en la Clasificación Internacional de Enfermedades de la OMS.
6. Tras la aparición del nuevo coronavirus, la Organización Mundial de la Salud convoca al comité de emergencias
7. Regulación, EECC y precio y reembolso: retos para mejorar el acceso a oncología de precisión en Europa
8. María Casanova-Acebes: "Mi grupo en el CNIO tratará de evitar la cronicidad del cáncer".
9. Nuevos datos de belantamab mafodotina para el tratamiento del MM refractar

10. La Enfermedad de Creutzfeldt-Jakob: cuando la incidencia se equipara a la prevalencia.
11. ASH 2020 consolida el beneficio clínico de Yescarta en LBCG y se abre paso en LNH
12. Del tratamiento sistémico al tópico, desafío en la psoriasis leve
13. Saturan hospitales en Jalisco por COVID-19.
14. Expertos advierten acerca de cuántos minutos diarios son suficientes para deprimirse al navegar en redes sociales.
15. Solares: todo lo nuevo para protegernos del sol
16. Coronavirus: Qué cuidados hay que tener al vacunar a personas alérgicas.
17. El estigma existente sobre salud mental hace la vida más difícil a los pacientes”.
18. La OMS declara emergencia de salud pública internacional el brote de ébola del Congo.
19. Salud amplía a los andaluces con diabetes el sistema flash de control de la glucosa.
20. Delhi reparte millones de máscaras tras declarar la emergencia de salud pública por contaminación.
21. El ébola, a un paso de ser una emergencia de salud pública internacional.
22. La crisis climática dañará toda su vida a un bebé que nazca hoy.
23. La dieta perfecta para salvar el planeta y la salud del ser humano.
24. Diabetes: bulos y mitos que te están intentado colar sobre esta enfermedad
25. Enfermedad de Crohn, el trastorno contra el que luchaba Sergio Gendler.
26. ¿Qué es el Parkinson?: Síntomas, causas y cómo tratar la enfermedad.
27. ¿Qué es el paludismo o malaria?: Síntomas, causas y cómo tratar la enfermedad.
28. Alzheimer: nueve cosas que puedes hacer para intentar evitarlo. [al-alzheimer-como-prevenir-evitar.html](#)
29. ¿Qué es la hipertensión arterial?: Síntomas, causas y cómo tratar la enfermedad.
30. ¿Qué es la hemofilia?: Síntomas, causas y cómo tratar la enfermedad.

TÍTULAR DE LOS ARTÍCULOS PARA EL GLOSARIO Y TEST DEL TEMA *POLÍTICA*.

1. Política de déficit cero y ayudas millonarias a los autónomos, así financia Alemania el cierre.
2. Otegi dice que el Supremo pretende “desestabilizar el escenario político”.
3. Los cuatro días de Errejón.
4. La guerra y la política a la carta.
5. El negacionismo en política.
6. La política digital no es blanca o negra, sino gris.
7. Rodrigo Rato: Cuando la política se estropea, lo demás también”.
8. Política Calígula.
9. Otorgar autoridad para anular el debate sobre política económica.
10. González Laya cree que la UE mantendrá el reconocimiento “político.”^a Guaidó pero no aclara con qué título.
11. La dieta perfecta para salvar el planeta y la salud del ser humano.

12. Los socios de Sánchez acusan al Supremo de «persecución política» cuando las causas no les favorecen.
13. El PNV denuncia una "persecución política".^{en} la causa abierta por el Supremo al consejero navarro Manuel Ayerdi por el 'caso Davalor'.
14. Del Cid: "Torremolinos se ha convertido en reflejo de la peor política, la que se olvida de la gente".
15. C-LM destaca el papel de la Política de Cohesión para garantizar la igualdad entre las diferentes regiones de Europa.
16. Mazón reivindica una política hídrica que evite la "guerra de regiones".
17. Asturias pide «calibrar y dotar adecuadamente» la nueva «herramienta de política industrial»
18. Federalismo, participación política, convivencia y municipalismo, algunas lecturas y reflexiones.
19. Telemadrid pide "voluntad política".^{al} Gobierno regional para solucionar los problemas de futuro en la cadena.
20. Inmigración en la UE: el 2020 termina sin avances en política de asilo.
21. Luisa Broto denuncia que "la Política Cultural de Zaragoza está en manos de Vox".
22. Montero receta "inteligencia política" a sus socios de Podemos tras los últimos choques.
23. Iturgaiz afea al Gobierno vasco que solo «corteje» a Podemos, «su socio en Madrid», con los Presupuestos
24. Indignación en Bruselas ante la deriva en política económica de Pedro Sánchez.
25. Putin felicita finalmente a Biden por su victoria en las elecciones.
26. ¿Nueva política social?: Los impactos sociales de la pandemia dejarán profundas huellas. El diseño de la política social post-covid debe arrancar.
27. Una dirigencia política desubicada.
28. Trump es la expresión de los nuevos tiempos que llegaron para quedarse en la política.
29. Albert Rivera: "Nunca he visto a nadie en política con el sentido de posesión del poder que tiene Pedro Sánchez".
30. Santa María del Páramo aprueba el presupuesto y reclama "altura política".

3. Extracción del glosario representativo

Para extraer los términos representativos de cada tema, se realizará una extracción terminológica de los documentos. Antes de ello, se debe eliminar previamente las palabras vacías (*stop words*).

Un proceso normal es el de tomar las palabras más frecuentes (pero con matices). Puesto que un término puede estar formado por una o varias palabras (por ejemplo . Si se toman solo términos de una sola palabra hay palabras que siendo muy frecuentes pueden dar un discriminante bajo. En este proyecto se ha elegido **utilizar términos simples** en vez de palabras compuestas por simplificación del proceso.

Se va a crear un proceso desde 0 para la generación del glosario. Se seguirán los **procedimientos fundamentales** para tareas relacionadas con la extracción automática de información:

- ♥ La tokenización.
- ♥ La normalización.
- ♥ La lematización y/o la *radicalización*.

La tokenización

En este primer proceso, se delimitan las palabras del texto, convirtiéndolas en elementos de una lista. Para este procedimiento, se hará uso de una **librería de Procesamiento de Lenguaje Natural llamada *spacy***. La librería soporta varios idiomas, entre ellos el español.

Al procesar el texto por las funciones de la librería, nos daremos cuenta de que están todas las palabras, incluidos los signos de puntuación y las palabras vacías.

En este mismo proceso, se realizará la **limpieza del texto**. *spacy* provee de funciones para eliminar los símbolos de puntuación y las *stopwords*. Tras observar las stopwords que provee *space*, se ha decidido crear un documento (*stop_words.txt*) en el cual se ha listado un total de 329 palabras que hemos considerado no relevantes dentro de un texto. A continuación se muestra un fragmento del contenido de las palabras en *stop_words.txt*:

(a, acá, ahí, ajena, ajenas, ajeno, ajenos, al, algo, algún, alguna, alguno, algunas, algunos, allá, allí, ambos, ante, antes, aquel, aquella, aquello, aquellas, aquellos, aquí, arriba, así, atrás, aun, aunque, bajo, bastante, bien, cabe, cada, casi, cierto, cierta, ciertos, ciertas, como, con, ...)

La normalización

Es importante que en el documento aparezcan las palabras siguiendo un consenso. En este proceso lo que se realizará será igualar el texto para no clasificar como diferentes palabras que solo cambian en su formato de escritura. Es decir, mantener un consenso para que las palabras *paciente* y *Paciente* no se consideren palabras diferentes. Para ello, se ha empleado la función `token.lower()` nativa de Python.

La lematización

La lematización relaciona una palabra derivada con su forma canónica o lema considerando la clase de palabra (adjetivo, verbo, sustantivo...). Se trata de un proceso que consume recursos (tiempo) y suele ser probabilístico, pudiendo dar resultados inesperados para algunos casos. La **lematización** funciona mejor cuando se desea procesar las palabras de manera similar a como lo hace un ser humano.

Tras varias pruebas para la creación del pre-glosario, **se ha decidido que la lematización daba mejores resultados**. Se ha empleado la función `token.lemma` del paquete *Natural Language Toolkit (NLTK)* para Python.

La radicalización

También llamado *stemming*, se trata del procedimiento de convertir palabras en raíces. Las raíces son la parte invariable de la palabra y, a cambio que con la lematización, las raíces generadas no tienen por qué ser palabras de un idioma. El *stemming* es mucho más rápido desde el punto de vista del procesamiento que la *lematización*. Puesto que extrae las raíces de las palabras, puede reconocer relaciones entre palabras de distinta clase.

- ♥ Si se recorta demasiado la raíz existe el problema de encontrar relaciones entre palabras que realmente no existen, proceso llamado *overstemming*.
- ♥ Si las raíces son demasiado extensas, puede ocurrir el problema de palabras que deberían compartir una misma raíz no lo hacen, proceso llamado *understemming*.

La **radicalización** o *stemming* se trata de una buena solución cuando la precisión no es muy importante y se requiere de un procesamiento eficiente. Para hacer el procesado de textos más rápido se ha optado por **aplicar la radicalización a los textos que vayan a ser clasificados**. Se ha empleado la función `token.stem` del paquete *Natural Language Toolkit (NLTK)* para Python.

3.1. Selección de los términos relevantes

Existen varios métodos para extraer los términos más relevantes para un texto, pero se ha considerado el **uso de la frecuencia para la extracción** una buena métrica. Se ha empleado la función `Counter.most_common` del paquete *Collections* para Python.

El proceso para discernir los términos relevantes no se encuentra optimizado puesto que se realizó un primer pre-procesamiento haciendo uso de las *stopwords* proporcionadas por el paquete *spacy*. Se puede observar en el apéndice D cómo la lista de *stopwords* de *spacy* no incluye la mayoría de artículos ni preposiciones ni conjunciones así como presenta conjunciones no comunes de verbos no auxiliares. Para resolver este problema, se ha creado una lista de *stopwords* propia en la cual no se han tenido en cuenta verbos auxiliares.

Una vez preprocesados los textos para el glosario, se guardan en un documento de texto plano las palabras más frecuentes para poder visualizarlo. Los glosarios finales consistirán en una lista de 30 palabras escogidas a mano a partir de la lista de las más frecuentes. No se escogerán las primeras más frecuentes necesariamente ya que, como se ha comentado antes, las *stopwords* que se han utilizado no incluyen verbos auxiliares. En el apéndice B se tiene el glosario completo de los términos por cada tema.

4. Clasificador de documentos

En esta sección se va a explicar cuales han sido los modelos y métricas utilizados para el desarrollo de este proyecto.

4.1. VSM

El modelo de **Vector Space Model (VSM)** consiste en crear un vector con palabras relevantes de los textos y utilizar una métrica que relacione estas palabras con los documentos. Existen muchas variantes de *VSM*, aunque el más popular es el modelo en el cual cada palabra tiene asociada su frecuencia en el documento. Con esa información se puede utilizar métricas como la *TF-IDF* para determinar relaciones entre documentos o *queries* y documentos.

Para este trabajo se han utilizado tres tipos de variantes de *VSM*, extraídas de dos librerías diferentes de procesamiento de lenguaje natural para Python:

- ♥ **CountVectorizer**, de la librería de *sklearn* crea un vector que contiene las tuplas de palabras y la frecuencia de las mismas en cada uno de los textos.
- ♥ **Doc2Bow**, de la librería *gensim* convierte un documento (una lista de palabras) en una Bolsa de Palabras (*Bag of Words*) compuesta por tuplas de palabras y la frecuencia de las mismas en cada uno de los textos.
- ♥ **Word2Vec**, de la librería *gensim* se encuentra disponible de dos formas: modelo *Skip-Gram* y modelo *Continuous Bag-Of-Words (CBOW)*. Para este proyecto se utilizará *Skip-Gram*, ya que los otros dos tipos de *VSM* que se van a utilizar son muy parecidos a *CBOW*. *Skip-gram* crea un vector con las palabras que aparecen en los textos y se asocian valores que representan la probabilidad de que aparezcan el resto de las palabras en su vecindad.

4.2. TF-IDF

TF-IDF, acrónimo de *Term Frequency times Inverse Document Frequency* (Frecuencia del Término por Frecuencia Inversa de Documento) se trata de una métrica dentro del ámbito de la Recuperación de la Información para encontrar el documento más relevante para cierto término dentro de un conjunto de documentos.

Para la aplicación del modelo *TF-IDF* (del paquete *gensim* para Python) se han seguido los siguientes pasos:

- ♥ Calcular la frecuencia teniendo en cuenta la extensión del propio documento.
- ♥ Ajustar la frecuencia del término según una escala logarítmica.
- ♥ Aplicar una normalización (dividiendo la frecuencia del término entre la frecuencia del término que más veces aparece en el texto).

Con estos procesos presentados, se puede introducir la ecuación 1. Muestra la multiplicación entre dos términos de frecuencia obteniendo un valor sin normalizar.

$$peso_{i,j} = frecuencia_{i,j} * \log_2 \frac{D}{frecuencia_documento_i} \quad (1)$$

dónde:

- $peso_{i,j}$ peso o relevancia que tiene un documento con respecto a cierto término.
- $frecuencia_{i,j}$ es la frecuencia del término localmente (en el documento actual).
- D es el número de documentos que componen el corpus.
- $frecuencia_documento_i$ es la frecuencia de ocurrencia del término en la colección de documentos.

Se ha decidido utilizar esta métrica para el proyecto ya que es una de las más comunes y fáciles de implementar, debido a la gran cantidad de ejemplos y librerías que se pueden encontrar.

4.3. Naïve Bayes

El modelo de clasificación de *Naïve Bayes* hace uso del teorema de Bayes para determinar cómo de probable es que un documento sea miembro de una categoría.

Se considera *Naïve* (ingenuo) porque considera la probabilidad de aparición de palabras independientes. Esta afirmación casi nunca es verdadera para los documentos a clasificar, puesto que la lengua siempre presenta reglas gramaticales, sintácticas, etc. Es decir, en la realidad existe correlación entre las palabras que aparecen en los textos.

En la ecuación 2 se puede observar el caso base de Bayes, se presenta cómo se halla la probabilidad de una hipótesis (H) dada una evidencia (E).

$$P(H | E) = P(H) \times \frac{P(E | H)}{P(E)} \quad (2)$$

dónde:

- $P(H | E)$ es la probabilidad de que ocurra una hipótesis (H) dada una evidencia (E).
- $P(H)$ es la probabilidad de que ocurra una hipótesis (H).
- $P(E | H)$ es la probabilidad de que ocurra un evento (E) dada una hipótesis (H).
- $P(E)$ es la probabilidad de que ocurra una evidencia (E).

Puesto que en este proyecto se están clasificando documentos, la *hipótesis* es si el documento pertenece a la clase C y la *evidencias* las palabras que aparecen en el documento. Puesto que el proceso de clasificación toma en cuenta dos o más hipótesis (entendiendo que hay dos o más clases en las cuales se clasifica), como se puede observar en la ecuación 3 se aplica el teorema de Bayes con forma de ratio.

$$\frac{P(C_1 | P)}{P(C_2 | P)} = P(C_1) \times \frac{P(P | C_1)}{P(P | C_2)} \quad (3)$$

dónde:

- $P(C_1 | P)$ es la probabilidad de que pertenezca a la clase 1 dada una palabra (P).
- $P(C_2 | P)$ es la probabilidad de que pertenezca a la clase 2 dada una palabra (P).
- $P(C_1)$ es la probabilidad de que ocurra la Clase 1.

- $P(P | C_1)$ es la probabilidad de que aparezca una palabra (P) dada una clase 2.
- $P(P | C_2)$ es la probabilidad de que aparezca una palabra (P) dada una clase 2.

Como un documento contiene un número mayor que 2 de palabras, la ecuación 3 se convierte en la ecuación 4

$$\frac{P(C_1 | W_1, W_2, W_N)}{P(C_2 | W_1, W_2, W_N)} = \frac{P(C_1) \times P(P_1 | C_1) \times P(P_2 | C_1) \times \dots \times P(P_N | C_1)}{P(C_2) \times P(P_1 | C_2) \times P(P_2 | C_2) \times \dots \times P(P_N | C_2)} \quad (4)$$

dónde:

- $P(C_1 | W_1, W_2, W_N)$ es la probabilidad de que pertenezca a la clase 1 dada unas palabras (W_i).
- $P(C_2 | W_1, W_2, W_N)$ es la probabilidad de que pertenezca a la clase 2 dada unas palabras (W_i).
- $P(C_1) \times P(P_1 | C_1) \times P(P_2 | C_1) \times \dots \times P(P_N | C_1)$ es la multiplicación de las probabilidades de que aparezca una palabra sabiendo que es de la Clase 1 multiplicado por la probabilidad de ocurrencia de la clase 1.
- $P(C_2) \times P(P_1 | C_2) \times P(P_2 | C_2) \times \dots \times P(P_N | C_2)$ es la multiplicación de las probabilidades de que aparezca una palabra sabiendo que es de la Clase 2 multiplicado por la probabilidad de ocurrencia de la clase 2.

4.4. Similaridad del coseno

Una vez se tiene el modelo de espacio vectorial y los elementos se han transformado en métricas de este mismo espacio. Se pueden comparar los documentos con distancias matemáticas, es decir aplicar una métrica que calcule el coseno del ángulo que existe entre los dos vectores. La ecuación 5 realiza el cálculo explicado para obtener la similitud entre dos vectores.

$$Similaridad_del_coseno = \frac{d \times d'}{|d| \times |d'|} \quad (5)$$

dónde:

- $d \times d'$ es la multiplicación de los vectores d y d' que representan dos elementos en el VSM (Modelo del espacio Vectorial).
- $|d| \times |d'|$ es la multiplicación de los valores absolutos de los vectores d y d' .

Se va a utilizar esta métrica tanto en el modelo de *TF-IDF* y para *Word2Vec*. Para el primer modelo se utilizará una matriz de similaridad de la librería de *gensim* la cual aplica la similaridad del coseno y para el segundo modelo se utilizará la función `cosine_similarity` de *sklearn* para comparar los vectores.

5. Implementación

En esta sección se desarrollarán todos los pasos a seguir para poder utilizar los programas realizados correctamente. El código se encuentra en el apéndice E ampliamente comentado.

El proyecto ha sido desarrollado y testeado en el entorno de Linux (Ubuntu 20.04.1 LTS), se recomienda realizar todos los pasos de la *Guía de Usuario* en un mismo entorno o similar. La programación se ha realizado en lenguaje *Python* (versión 3.8.5).

Nota para el lector: Para moverse a una carpeta se utiliza `cd nombre_carpeta`, se utiliza `cd ..` para ir atrás en la carpeta.

5.1. Instalación para Windows

Opción 1: Descarga de una máquina virtual.

Descargar una máquina virtual, por ejemplo VMWare (<https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html>), descargar una imagen de Linux (por ejemplo Ubuntu) (<https://ubuntu.com/download/desktop>) y seguir los pasos de instalación (ejemplo: <https://platzi.com/tutoriales/1050-programacion-basica/295-instalar-linux-ubuntu-1404-como-una-maquina-virtual-vmware-player/>).

A continuación abrir la línea de comandos, ir a la carpeta *Document_classification* y realizado los pasos de la sección 5.3.2.

Opción 2: Uso de Anaconda en Windows.

Descargar Anaconda de la página oficial (<https://www.anaconda.com/products/individual#Downloads>) e instalar. A continuación abrir la línea de comandos, ir a la carpeta *Document_classification* y realizado los pasos de la sección 5.3.2.

5.2. Instalación para Ubuntu (Linux)

Verificar que se tiene instalado Python 3, ir a la carpeta *Document_classification* e instalar las dependencias siguiendo los pasos de la sección 5.3.2.

5.3. Guía de Usuario

5.3.1. Estructura del proyecto

La carpeta del proyecto se llama *Document_classification*. Contiene los elementos que se listarán a continuación:

♥ Carpeta Documentos.

- Carpeta **Deportes**. Contiene 30 archivos de texto del tema deportes.
- Carpeta **Política**. Contiene 30 archivos de texto del tema política.
- Carpeta **Salud**. Contiene 30 archivos de texto del tema salud.
- Archivo **paginas_web_docs.txt**. Archivo de texto con el listado con todas las páginas web de donde se han extraído los documentos.

♥ Carpeta Glosario

- Archivo **Deportes_glosario.txt**. Archivo de texto con el glosario escogido para el tema deportes.
- Archivo **Política_glosario.txt**. Archivo de texto con el glosario escogido para el tema política.
- Archivo **Salud_glosario.txt**. Archivo de texto con el glosario escogido para el tema salud.

♥ Carpeta Pre-Glosario

- Archivo **Deportes.txt**. Archivo de texto con la frecuencia de las palabras para el tema deportes.
- Archivo **Política.txt**. Archivo de texto con la frecuencia de las palabras para el tema política.
- Archivo **Salud.txt**. Archivo de texto con la frecuencia de las palabras para el tema salud.

♥ Carpeta Resultados

- Carpeta **Naive-Bayes**.
 - Archivo **naivesbayes_results.txt**. Archivo de texto que contiene una matriz relevancias de N documentos x M clases de aplicando el modelo *Naïve Bayes* para clasificar.
- Carpeta **tfidf**.

- Archivo **tfidf_Deportes_glosario.txt**. Archivo de texto que contiene una matriz de relevancias para los N documentos al aplicar el modelo *tfidf* con el glosario de deportes.
- Archivo **tfidf_Politica_glosario.txt**. Archivo de texto que contiene una matriz de relevancias para los N documentos al aplicar el modelo *tfidf* con el glosario de política.
- Archivo **tfidf_Salud_glosario.txt**. Archivo de texto que contiene una matriz de relevancias para los N documentos al aplicar el modelo *tfidf* con el glosario de salud.
- Carpeta **word2vec**.
 - Archivo **word2vec_Deportes_glosario.txt**. Archivo de texto que contiene una matriz de relevancias para los N documentos al aplicar el modelo *word2vec* con el glosario de deportes.
 - Archivo **word2vec_Politica_glosario.txt**. Archivo de texto que contiene una matriz de relevancias para los N documentos al aplicar el modelo *word2vec* con el glosario de deportes.
 - Archivo **word2vec_Salud_glosario.txt**. Archivo de texto que contiene una matriz de relevancias para los N documentos al aplicar el modelo *word2vec* con el glosario de deportes.
- Carpeta **Clasificacion**.
 - Archivo **NaiveBayes.txt**. Archivo de texto que contiene la predicción de la clasificación del documento con su clase y el valor real al aplicar el modelo *Naïve Bayes*.
 - Archivo **tfidf.txt**. Archivo de texto que contiene la predicción de la clasificación del documento con su clase y el valor real al aplicar el modelo *tfidf*.
 - Archivo **word2vec.txt**. Archivo de texto que contiene la predicción de la clasificación del documento con su clase y el valor real al aplicar el modelo *word2vec*.
- Carpeta **Evaluacion**.
 - Archivo **NaiveBayes.txt**. Archivo de texto con la exactitud, la precisión y sensibilidad/exhaustividad del modelo *Naïve Bayes*.
 - Archivo **tfidf.txt**. Archivo de texto con la exactitud, la precisión y sensibilidad/exhaustividad del modelo *tfidf*.
 - Archivo **word2vec.txt**. Archivo de texto con la exactitud, la precisión y sensibilidad/exhaustividad del modelo *word2vec*.
- ♥ Ejecutable **clasificador.py**
- ♥ Ejecutable **crear_corpus.py**
- ♥ Ejecutable **evaluator.py**
- ♥ Ejecutable **lector.py**
- ♥ Archivo **stop_words.txt**. Archivo de texto con el listado de las *stop_words* seleccionadas e introducidas a mano.
- ♥ Ejecutable **Dependencias**. Instalación automática de los paquetes necesarios para el correcto funcionamiento del proyecto.

5.3.2. Dependencias

Las dependencias y paquetes a instalar son los siguientes:

```
$ sudo apt-get install python3-tk
$ pip3 install pandas
$ pip3 install spacy
$ pip3 install gensim
$ pip3 install nltk
$ pip3 install sklearn
$ pip3 install matplotlib
$ pip3 install seaborn
```

Para instalar automáticamente los paquetes necesarios escribir el comando:

```
$ python3 Dependencias.py
```

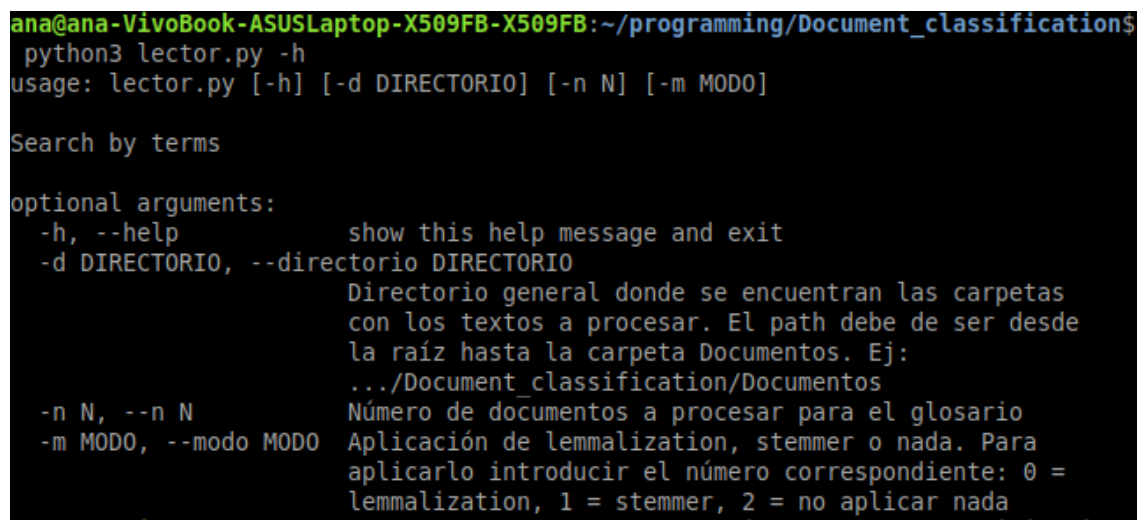
A la hora de instalar las dependencias del programa, el programa pedirá la contraseña del perfil del usuario (al usar poderes de super usuario - sudo).

5.4. Flujo de trabajo (*Workflow*) del proyecto

El orden de uso de los ejecutables será: `lector.py` > `crear_corpus.py` > `clasificador.py` > `evaluador.py`

♥ Paso 1: Uso de `lector.py`

El script `lector.py` se utiliza para crear el listado previo de palabras que comprenderán el pre-glosario. Como se puede observar en la Figura 2, mediante el comando `lector.py -h` se puede obtener ayuda sobre la sintaxis.



```
ana@ana-VivoBook-ASUSLaptop-X509FB-X509FB:~/programming/Document_classification$
python3 lector.py -h
usage: lector.py [-h] [-d DIRECTORIO] [-n N] [-m MODO]

Search by terms

optional arguments:
  -h, --help            show this help message and exit
  -d DIRECTORIO, --directorio DIRECTORIO
                        Directorio general donde se encuentran las carpetas
                        con los textos a procesar. El path debe de ser desde
                        la raíz hasta la carpeta Documentos. Ej:
                        ../Document_classification/Documentos
  -n N, --n N           Número de documentos a procesar para el glosario
  -m MODO, --modo MODO  Aplicación de lemmalization, stemmer o nada. Para
                        aplicarlo introducir el número correspondiente: 0 =
                        lemmalization, 1 = stemmer, 2 = no aplicar nada
```

Figura 2: Comando de ayuda del creador del glosario

Por tanto, para ejecutar el programa se necesita introducir por terminal el *path* desde la raíz hasta donde se encuentra la carpeta *Document_classification*, el número de documentos a utilizar para la generación del pre - glosario y elegir *lemmalization* como modo. Ejemplo:

```
$ python3 lector.py -d <raiz_hasta_carpeta>/Document_classification -n 15 -m 0
```

lector.py se encarga de recibir datos por la línea de comandos y de leer los archivos de texto para generar los pre-glosarios. Este script llama a funciones de **crear_corpus.py**, el cual se encarga de pre-procesar los documentos (aplicar la tokenización, la normalización y la lemmalización). En este pre-procesamiento se ha utilizado el archivo `stop_words.txt` para eliminar las palabras de parada.

Se almacenará el resultado, una lista de tuplas compuesta por los tokens y su frecuencia, en la carpeta *Pre-Glosario* como texto plano.

♥ Paso 2: Selección del glosario

Se debe extraer a mano los 30 elementos para los glosarios documentales de los archivos de texto de la carpeta *Pre-Glosario*. Aunque es posible filtrar únicamente los primeros términos, es recomendable obtener los valores manualmente para generar un glosario que pueda albergar términos que abarquen los conocimientos generales del tema. En el apéndice C se tiene como se ha realizado la selección con los archivos del *Pre-Glosario*.

Los glosarios finales deben de almacenarse en la carpeta *Glosario* como archivo de texto con los términos almacenados separados por espacios en una misma línea. Existen tres archivos de texto (uno por tema) con los glosarios que se han creado a mano.

♥ Paso 3: Uso de `clasificador.py`

Una vez se tiene el glosario, se utiliza `clasificador.py` para crear un modelo y clasificar los documentos. En este ejecutable, se encuentran los tres modelos que se han creado para la clasificación documental. Se puede observar en la Figura 3 el uso del comando `clasificador.py -h` para obtener ayuda sobre la sintaxis.

```
ana@ana-VivoBook-ASUSLaptop-X509FB-X509FB:~/programming/Document_classification$
python3 clasificador.py -h
usage: clasificador.py [-h] [-d DIRECTORIO] [-n NMIN] [-r RANGO] [-m MODELO]

Search by terms

optional arguments:
  -h, --help            show this help message and exit
  -d DIRECTORIO, --directorio DIRECTORIO
                        Directorio general donde se encuentran las carpetas
                        con los textos a procesar. El path debe de ser desde
                        la raíz hasta la carpeta Documentos. Ej:
                        ../Document_classification/Documentos
  -n NMIN, --nmin NMIN  Posición a partir de la cual se utilizarán los
                        documentos (Por ejemplo, a partir del documento 16)
  -r RANGO, --rango RANGO
                        Número de documentos totales a utilizar para la
                        clasificación de test (Por ejemplo si deseamos 15
                        documentos, del 16 al 30)
  -m MODELO, --modelo MODELO
                        Modelo a utilizar para el clasificador. 0 = VSM con
                        tf-idf, 1 = VSM (word2vec), 2 = Naive Bayes
```

Figura 3: Comando de ayuda del clasificador

Para utilizar correctamente el programa, se presenta un ejemplo a continuación.

- ♥ El **comando -n** indica a partir de qué documento se analizará para la clasificación.
- ♥ El **comando -r** indica el número de documentos que se analizarán para la clasificación (Total_documentos - Posición_Documento_inicial).
- ♥ El **comando -m** indica que modelo se desea utilizar.

```
$ python3 clasificador.py -d <raiz_hasta_carpeta>/Document_classification -n 16 -r 15
-m 0
```

Por defecto, **los valores de n , r y m se encuentran seleccionados como 16, 15 y 0** respectivamente. Por lo que si se desean esos parámetros, no es necesario introducir valores específicos.

Los modelos utilizados han sido:

1. Doc2Bow con TF-IDF con matriz de similaridad del coseno de *gensim*. Se selecciona por línea de comando con `-m 0`.
2. Word2Vec de *gensim* con `cosine_similarity` de *sklearn*. Se selecciona por línea de comando con `-m 1`.
3. CountVectorizer con Naïve-Bayes de *sklearn*. Se selecciona por línea de comando con `-m 2`

`clasificador.py` se encarga de crear un VSM para los documentos a partir del glosario e implementar las métricas para clasificar los documentos con respecto a los glosarios generados a mano. Todos los resultados se almacenan en la carpeta *Resultados*. En la carpeta *Clasificación* se guarda una matriz con el nombre del documento, el valor del tema al que pertenece realmente el documento, la probabilidad de predicción correcta calculada y el valor de la clasificación predicha.

Para calcular la bondad de las predicciones, se llama automáticamente al script `evaluator.py`. Se encarga de calcular la *precisión*, *exactitud* y *sensibilidad/exhaustividad* del método utilizado, los imprime por pantalla y muestra un mapa de calor que representa la *matriz de confusión* de la clasificación de los documentos (éstos mapas se muestran en la sección 6).

6. Resultados y Evaluación de los métodos

En esta sección se van a mostrar los resultados obtenidos y se van a comparar los distintos modelos utilizados. En los dos primeros modelos, se ha calculado la probabilidad de que el documento pertenezca a cada una de las clases. Se generan tres archivos (uno por tema) que contienen la probabilidad de que un documento pertenezca a una clase y se guardan en la carpeta de *Resultados* (en la carpeta con el nombre del algoritmo utilizado). En el caso de *Naïve-Bayes*, se obtiene directamente un único documento con el valor del tema al que predice que pertenece el documento.

Para tomar la decisión de clasificación de un documento a un tema (a excepción del modelo de *Naïve-Bayes*), se comparan las tres probabilidades y se asigna el documento al tema para el cual el valor sea más alto. Se ha decidido realizar este método por simplificación del proceso aunque pueda conllevar fallos. Por ejemplo, si para un documento se tiene la misma probabilidad para dos temas, se cogerá el último tema que se haya comparado.

Naïve-Bayes requiere que las clases sean de tipo numérico, por tanto, para poder realizar una comparación entre los diferentes modelos y para un mejor procesamiento, se han empleado los mismos criterios en todos los modelos para la obtención de los resultados. Es decir, a cada tema se le ha asignado un valor numérico:

♥ Deportes → 0.

♥ Política → 1.

♥ Salud → 2.

6.1. Cálculo de la bondad (Precisión, Exactitud, Sensibilidad/Exhaustividad y valor F1)

Para el cálculo de la bondad se ha aplicado los parámetros de precisión (ecuación 6), exactitud (ecuación 7), sensibilidad/exhaustividad (ecuación 8) y valor F1 (ecuación 9).

- ♥ **Precisión** indica, de las clasificaciones realizadas para cada clase, ¿cuántas han sido correctas?
- ♥ **Exactitud** indica, de las clasificaciones totales realizadas, ¿cuántas han sido correctas?
- ♥ **Exhaustividad** indica, de los documentos clasificados para una clase, ¿cuántos están correctamente clasificados?
- ♥ **Valor F1** se trata de una combinación de las medidas de precisión y de exhaustividad en un sólo valor para comparar el rendimiento combinado de ambos valores fácilmente.

$$Precision = \frac{VP}{VP + FP} \quad (6)$$

$$Exactitud = \frac{VP + VN}{T} \quad (7)$$

$$Sensibilidad = \frac{VP}{VP + FN} \quad (8)$$

$$F1 = 2 \times \frac{Precision \times Exhaustividad}{Precision + Exhaustividad} \quad (9)$$

dónde:

- VP son los **verdaderos positivos**, los documentos clasificados en una clase correctamente.
- VN son los **verdaderos negativos**, los documentos clasificados en una clase pero no pertenecen a esa clase.
- FP son los **falsos positivos**, los documentos no clasificados en una clase pero sí pertenecen a esa clase.
- FN son los **falsos negativos**, los documentos no clasificados en una clase y no pertenecen a esa clase.
- T son las **observaciones totales** (número total de documentos).

6.2. Clasificación con el modelo Doc2Bow y la métrica TFIDF y similaridad del coseno

La clasificación final se encuentra en la Figura 4 como una matriz de confusión. Ésta matriz presenta en el eje X las predicciones que ha realizado el modelo y en el eje Y los valores reales. El modelo ha clasificado correctamente a los documentos en la clase deportes y la mayoría de documentos de salud en su clase correspondiente, errando en clasificar correctamente varios documentos de política.

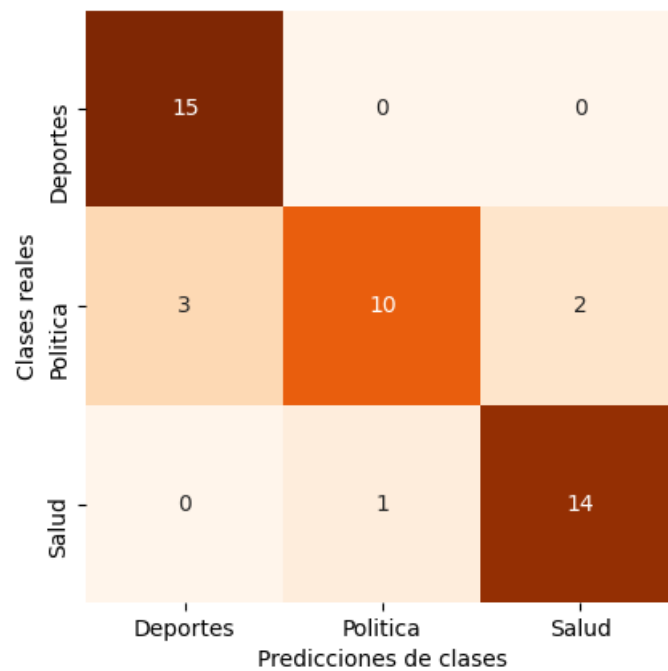


Figura 4: Matriz de confusión con mapa de calor representando la clasificación de los documentos acorde a la clasificación realizada con TFIDF

Se puede observar en el Cuadro 1 las probabilidades obtenidas para cada tema para cada uno de los documentos. En rosa se señalan las probabilidades máximas y, por tanto, la clase a la que se ha predicho que pertenece el documento.

Los resultados obtenidos muestran que el glosario de política no se encuentra totalmente disjunto con respecto al resto de glosarios. Este problema puede ser debido a que en la actualidad política y en el rango de tiempo escogido de los documentos (años 2019 - 2020) el tema sanitario se trata de un problema acuciante y presente constantemente.

Documento	Deportes	Política	Salud
<i>deportes16</i>	0.3178	0.1217	0.0131
<i>deportes17</i>	0.3863	0.0693	0.0
<i>deportes18</i>	0.1659	0.0390	0.0
<i>deportes19</i>	0.3006	0.0236	0.0
<i>deportes20</i>	0.4389	0.0	0.0037
<i>deportes21</i>	0.2456	0.1077	0.1128
<i>deportes22</i>	0.2636	0.0	0.0
<i>deportes23</i>	0.2143	0.0382	0.0
<i>deportes24</i>	0.2246	0.0	0.0
<i>deportes25</i>	0.1844	0.0436	0.0
<i>deportes26</i>	0.1655	0.0999	0.0
<i>deportes27</i>	0.3187	0.0	0.0
<i>deportes28</i>	0.2892	0.0	0.0381
<i>deportes29</i>	0.2916	0.0	0.0844
<i>deportes30</i>	0.2995	0.0465	0.0569
<i>politica16</i>	0.1357	0.0988	0.0
<i>politica17</i>	0.1346	0.1342	0.0
<i>politica18</i>	0.0	0.2052	0.1291
<i>politica19</i>	0.0	0.0	0.0
<i>politica20</i>	0.0187	0.1757	0.0
<i>politica21</i>	0.0164	0.1279	0.0
<i>politica22</i>	0.0	0.5212	0.0
<i>politica23</i>	0.0480	0.2148	0.2205
<i>politica24</i>	0.0187	0.3314	0.0
<i>politica25</i>	0.0778	0.0	0.0
<i>politica26</i>	0.0	0.1140	0.0470
<i>politica27</i>	0.0288	0.2698	0.1255
<i>politica28</i>	0.1051	0.1462	0.0
<i>politica29</i>	0.0	0.21429	0.0
<i>politica30</i>	0.0	0.2935	0.0
<i>salud16</i>	0.01442	0.0	0.3285
<i>salud17</i>	0.6902	0.0780	0.0957
<i>salud18</i>	0.0127	0.0436	0.1649
<i>salud19</i>	0.0518	0.0	0.2539
<i>salud20</i>	0.0428	0.0781	0.05661
<i>salud21</i>	0.0382	0.0356	0.1046
<i>salud22</i>	0.0266	0.0053	0.1529
<i>salud23</i>	0.0441	0.0	0.1449
<i>salud24</i>	0.0334	0.0	0.2617
<i>salud25</i>	0.0171	0.0	0.3008
<i>salud26</i>	0.0031	0.0	0.2697
<i>salud27</i>	0.0	0.0	0.2681
<i>salud28</i>	0.02049	0.0676	0.2362
<i>salud29</i>	0.0	0.0	0.2105
<i>salud30</i>	0.0	0.0	0.3484

Cuadro 1: TFIDF

6.3. Clasificación con el modelo Word2Vec y la métrica de la similitud del coseno

La clasificación final se encuentra en la Figura 5 como una matriz de confusión. Ésta matriz presenta en el eje X las predicciones que ha realizado el modelo y en el eje Y los valores reales. El modelo no ha conseguido clasificar correctamente por completo ninguna clase y ha errado mayoritariamente clasificando en la clase salud documentos pertenecientes a deportes o política.

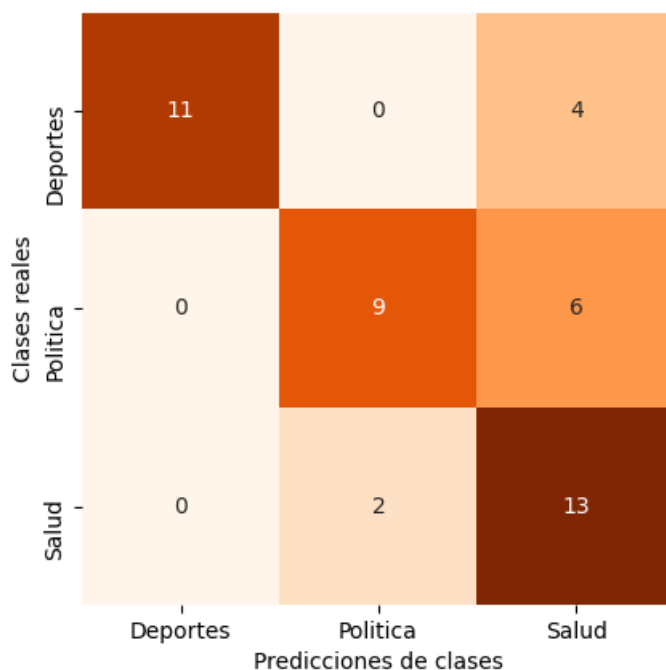


Figura 5: Matriz de confusión con mapa de calor representando la clasificación de los documentos acorde a la clasificación realizada con Word2Vec

En el Cuadro 2 se muestran las probabilidades obtenidas para cada tema para cada uno de los documentos. En rosa se señalan las probabilidades máximas y, por tanto, la clase a la que se ha predicho que pertenece el documento.

Los resultados obtenidos pueden deberse a la forma en la cual se ha aplicado el glosario al modelo. El modelo Word2Vec realiza una predicción sintáctica de las palabras en la vecindad, pero al entrenar nuestro modelo con un glosario compuesto por palabras con sentido semántico en vez de sintáctico no se extrae el potencial de uso del modelo.

Debido a que las predicciones que realiza en cada ejecución son probabilísticas, si se vuelve a ejecutar el algoritmo es probable que se consigan otros resultados.

Documento	Deportes	Política	Salud
<i>deportes16</i>	-0.3314	-0.2228	0.0851
<i>deportes17</i>	0.5514	-0.2861	0.0613
<i>deportes18</i>	0.7379	-0.1382	0.1066
<i>deportes19</i>	0.5864	0.3386	-0.2253
<i>deportes20</i>	0.6185	-0.2347	-0.010
<i>deportes21</i>	0.0862	0.1200	0.4155
<i>deportes22</i>	0.0900	-0.0092	0.1222
<i>deportes23</i>	0.7063	-0.0435	0.03632
<i>deportes24</i>	0.3634	0.1646	-0.0871
<i>deportes25</i>	0.4459	-0.040	-0.1604
<i>deportes26</i>	0.2837	-0.2763	-0.0664
<i>deportes27</i>	0.4042	-0.3017	-0.5690
<i>deportes28</i>	0.6952	0.0133	0.3061
<i>deportes29</i>	0.1836	-0.3001	0.4636
<i>deportes30</i>	0.4284	0.2712	-0.1272
<i>politica16</i>	0.2980	0.5305	-0.0206
<i>politica17</i>	-0.1197	0.2839	0.0639
<i>politica18</i>	0.0524	0.1036	0.0268
<i>politica19</i>	0.0	0.0	0.0
<i>politica20</i>	-0.3952	0.3796	0.1797
<i>politica21</i>	-0.1115	0.0346	-0.1015
<i>politica22</i>	-0.3029	0.5315	0.2804
<i>politica23</i>	-0.0436	0.3634	0.5421
<i>politica24</i>	-0.4145	0.5099	0.0403
<i>politica25</i>	-0.1250	0.3733	0.1798
<i>politica26</i>	-0.6532	-0.2414	0.0350
<i>politica27</i>	-0.1287	-0.2584	0.4168
<i>politica28</i>	-0.3771	-0.3624	0.1096
<i>politica29</i>	0.0261	0.3279	-0.0292
<i>politica30</i>	-0.4514	-0.1343	0.098
<i>salud16</i>	0.2035	0.1514	0.3174
<i>salud17</i>	-0.4382	-0.0987	0.3249
<i>salud18</i>	-0.0672	0.4353	0.3103
<i>salud19</i>	-0.1309	0.1681	0.4792
<i>salud20</i>	0.0700	0.7549	-0.0932
<i>salud21</i>	-0.0736	0.3973	0.4648
<i>salud22</i>	-0.1391	0.3428	0.5809
<i>salud23</i>	0.0728	0.1946	0.4053
<i>salud24</i>	0.0087	0.2653	0.6164
<i>salud25</i>	-0.1882	0.1778	0.5481
<i>salud26</i>	-0.1088	0.3819	0.5542
<i>salud27</i>	-0.0960	0.2761	0.6601
<i>salud28</i>	-0.3520	0.0248	0.5590
<i>salud29</i>	-0.1861	0.2810	0.5602
<i>salud30</i>	-0.0221	0.1460	0.5741

Cuadro 2: Word2Vec

6.4. Clasificación con el modelo CounterVectorizer y la métrica de Naïve Bayes

La clasificación final se encuentra en la Figura 6 como una matriz de confusión. Ésta matriz presenta en el eje X las predicciones que ha realizado el modelo y en el eje Y los valores reales. Se aprecia que el modelo ha clasificado exitosamente todos los documentos pertenecientes a las clases deportes y salud pero ha errado al clasificar 3 elementos de política en la clase deportes.

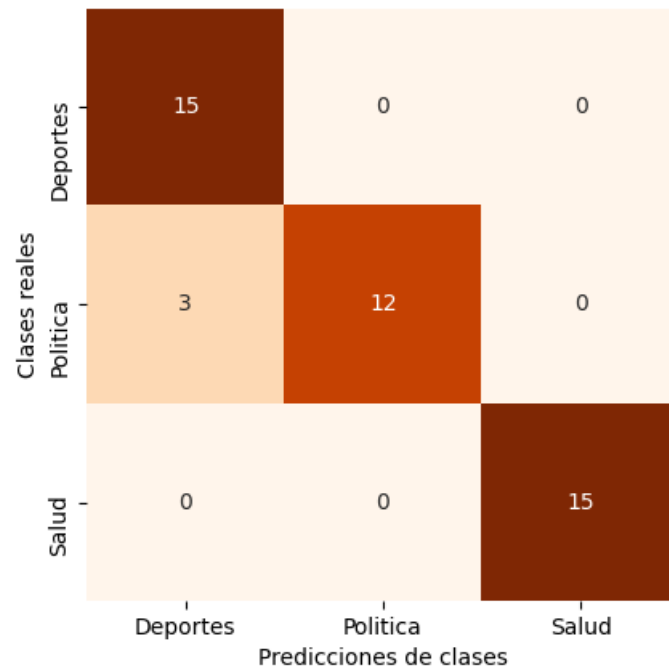


Figura 6: Matriz de confusión con mapa de calor representando la clasificación de los documentos acorde a la clasificación realizada con Naïve Bayes

Como se explicó en la sección 5, la implementación de *Naïve-Bayes* sólo nos proporciona el valor de la probabilidad (que calcula el algoritmo internamente) de el documento pertenezca al tema en el cual ha sido clasificado. Por ello, en el Cuadro 3 se muestran las probabilidades obtenidas y el valor del tema que ha calculado este algoritmo. En gris se señalan los que el algoritmo ha clasificado de manera errónea.

Como se puede observar, en general el algoritmo da bastantes buenos resultados. Como en el caso de TF-IDF, parece que donde falla el algoritmo es en la clasificación de documentos de política. Como se explicó antes, probablemente parte del problema se puede deber a que el glosario de este tema no contiene suficientes palabras disjuntas con los otros temas.

Documento	Probabilidad	Clase
<i>deportes16</i>	0.9952	0
<i>deportes17</i>	0.9971	0
<i>deportes18</i>	0.9560	0
<i>deportes19</i>	0.9985	0
<i>deportes20</i>	1.0	0
<i>deportes21</i>	0.9380	0
<i>deportes22</i>	0.9995	0
<i>deportes23</i>	0.9952	0
<i>deportes24</i>	0.9999	0
<i>deportes25</i>	0.9155	0
<i>deportes26</i>	0.9980	0
<i>deportes27</i>	0.9981	0
<i>deportes28</i>	0.9841	0
<i>deportes29</i>	0.8989	0
<i>deportes30</i>	0.8916	0
<i>politica16</i>	0.4013	0
<i>politica17</i>	0.4719	1
<i>politica18</i>	0.6231	1
<i>politica19</i>	0.3333	0
<i>politica20</i>	0.9904	1
<i>politica21</i>	0.9560	1
<i>politica22</i>	0.9999	1
<i>politica23</i>	0.6945	1
<i>politica24</i>	0.9971	1
<i>politica25</i>	0.5010	0
<i>politica26</i>	0.8189	1
<i>politica27</i>	0.8477	1
<i>politica28</i>	0.9348	1
<i>politica29</i>	0.9422	1
<i>politica30</i>	0.9997	1
<i>salud16</i>	0.9999	2
<i>salud17</i>	0.8697	2
<i>salud18</i>	0.9967	2
<i>salud19</i>	0.9996	2
<i>salud20</i>	0.4856	2
<i>salud21</i>	0.9743	2
<i>salud22</i>	0.9999	2
<i>salud23</i>	0.9976	2
<i>salud24</i>	0.9999	2
<i>salud25</i>	1.0	2
<i>salud26</i>	1.0	2
<i>salud27</i>	0.9999	2
<i>salud28</i>	0.9999	2
<i>salud29</i>	0.9999	2
<i>salud30</i>	0.9999	2

Cuadro 3: Naïve Bayes

6.5. Comparativa

En el siguiente apartado se muestra una comparativa de los cuatro valores escogidos para evaluar la bondad de los modelos (Cuadro 4). A excepción del *valor F*, los resultados se han obtenido mediante el paquete *scikit* para *Python*. Puesto que se trata de una clasificación multiclase, se ha empleado el comando `average: 'macro'`, el cual calcula las métricas para clase y realiza la media sin pesos (no se da importancia a ninguna clase).

Modelo	Precisión	Exactitud	Exhaustividad	Valor F
<i>TF-IDF</i>	0.8724	0.8666	0.8666	0.8695
<i>Word2Vec</i>	0.7944	0.7333	0.7333	0.7626
<i>Naïve-Bayes</i>	0.9444	0.9333	0.9333	0.9388

Cuadro 4: Comparativa de los valores de bondad (precisión, exactitud y exhaustividad) para los tres modelos empleados.

Los valores del valor F, se han calculado manualmente (mostrados a continuación) y serán útiles para la evaluación de los modelos.

$$F1_{TFIDF} = 2 \times \frac{0.8724 \times 0.8666}{0.8724 + 0.8666} = 0,8695$$
$$F1_{Word2Vec} = 2 \times \frac{0.7944 \times 0.7333}{0.7944 + 0.7333} = 0,7626$$
$$F1_{NB} = 2 \times \frac{0.9444 \times 0.9333}{0.9444 + 0.9333} = 0,9388$$

Se ha realizado el cálculo del valor F puesto que es una medida que considera la precisión y la exhaustividad. Su rango se encuentra entre 0 y 1 y cuanto más cerca de la unidad se encuentre mejor se considera un modelo. Tras analizar visualmente los resultados de los Cuadros 4, 5 y 6 y extraer analíticamente los valores de F1 mediante la ecuación 9 se ha concluido que el ranking de la funcionalidad de los clasificadores documentales (en orden descendente) es el siguiente:

1. Naïve Bayes
2. Doc2Vec con tfidf
3. Word2Vec

7. Conclusiones

En este proyecto se ha diseñado un sistema de clasificación documental para tres temas específicos. Aunque los temas nos parecían disjuntos, es posible que por culpa de la situación actual, los documentos compartían más similitudes de las que se pensaban.

En primer lugar, la selección a mano de los documentos fue tediosa. Además, muchos artículos encontrados hablaban de la pandemia que estamos sufriendo. Se intentó solventar esto encontrando artículos más variados yendo un poco atrás en el tiempo en las fechas de las noticias.

En cuanto al glosario, realizarlo prácticamente a mano no es muy estándar, ya que puede generar sesgos dependiendo de la persona que realiza el glosario. Es por ello que tal vez definiendo unas *stopwords* mejores se pudiese automatizar mejor el sistema, teniendo que escoger tan solo las palabras más frecuentes.

Los algoritmos escogidos han dado resultados variados. El problema principal del primer algoritmo utilizado, (Doc2Bow con TFIDF y similitud del coseno) es que la simplicidad de su métrica no ofrece mucha información del documento. Aún así ha demostrado dar unos resultados bastante óptimos. Cabe observar que todos los valores de similitud son bastante bajos en general, incluso cuando acierta la clasificación. El segundo algoritmo (Word2Vec con similitud del coseno) es el que peor resultados presenta. Esto se debe a que el vector contiene predicciones de las palabras siguientes y debido a que los glosarios son palabras que no forman frases las predicciones que realiza el modelo no pueden ser buenas para el glosario. Además, como se trata de un algoritmo probabilístico en cada ejecución del programa dará resultados diferentes. Por último, el algoritmo de Naïve-Bayes, ha demostrado dar unos resultados bastante óptimos, fallando tan sólo

en los documentos de política. Si se observan los valores de de probabilidad, de que el modelo pertenezca a ese tema, tanto en deportes como en salud estos son bastante altos. En cambio, en los de política se observa que el modelo no es capaz de asegurar en más de un 50 % que sea correcto. Esto no hace más que afianzar la hipótesis de que el glosario de política no esté bien diseñado.

Aún así no se puede descartar la posibilidad de que los documentos escogidos, en especial los de política, tengan características que hagan que su clasificación sea especialmente difícil. En un análisis superficial de los textos no hemos conseguido discernir que hace que estos documentos sean mal catalogados. Es por ello que la hipótesis principal es que el glosario de política debe de tener algún defecto.

Podemos concluir que el trabajo presentado nos ha proporcionado una visión general de la complejidad que supone desarrollar buenos algoritmos de clasificación. No es tan solo importante que el algoritmo sea bueno, si no que los parámetros proporcionados a dichos algoritmos estén bien diseñados. En este caso, dichos parámetros serían el glosario y los documentos que entraran en el sistema para ser clasificados.

Bibliografía

- [1] *Code Faster with Line-of-Code Completions, Cloudless Processing*. URL: <https://www.kite.com> (visitado 22-12-2020).
- [2] Kelly Epley. *Naive Bayes Document Classification in Python*. Medium. 30 de jun. de 2019. URL: <https://towardsdatascience.com/naive-bayes-document-classification-in-python-e33ff50f937e> (visitado 22-12-2020).
- [3] *Gensim: topic modelling for humans*. URL: <https://radimrehurek.com/gensim/models/word2vec.html> (visitado 22-12-2020).
- [4] *sklearn.feature_extraction.text.CountVectorizer — scikit-learn 0.23.2 documentation*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html (visitado 22-12-2020).

A. Listado de los documentos utilizados

♥ DOCUMENTOS PARA EL GLOSARIO DE DEPORTES.

1. **Titular:** *El jugador de rugby de 42 años que no recuerda haber ganado el Mundial.*
Fuente digital: El País.
Autor: Luis Javier González.
Fecha de Publicación: 13 / Diciembre / 2020.
Url: <https://elpais.com/deportes/2020-12-13/el-jugador-de-rugby-de-42-anos-que-no-recuerda-haber-ganado-el-mundial.html>
2. **Titular:** *Barcelona - PSG y Atalanta - Real Madrid y Atleti - Chelsea, duelos de octavos de final en la Champions.*
Fuente digital: El Español.
Autor: Jose Nieto.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: https://www.lespanol.com/deportes/futbol/20201214/streaming-directo-sorteo-octavos-final-champions-league/543446000_0.html
3. **Titular:** *Gervasio Deferr: "El problema es que los que mandan no han hecho deporte en su vida".*
Fuente digital: El Periódico.
Autor: Roger Pascual.
Fecha de Publicación: 11 / Diciembre / 2020.
Url: <https://www.elperiodico.com/es/deportes/20201211/gervasio-deferr-problema-mandan-han-11374467>
4. **Titular:** *El viejo truco que funciona otra vez a Zidane.*
Fuente digital: El Mundo.
Autor: Miguel A. Herguedas.
Fecha de Publicación: 13 / Diciembre / 2020.
Url: <https://www.elmundo.es/deportes/futbol/primera-division/2020/12/13/5fd68c3721efa013058b460a.html>
5. **Titular:** *Messi sufre por el Barça.*
Fuente digital: El País.
Autor: Ramon Besa.
Fecha de Publicación: 13 / Diciembre / 2020.
Url: <https://elpais.com/deportes/2020-12-13/messi-sufre-por-el-barca.html>
6. **Titular:** *El deporte federado vuelve a las canchas.*
Fuente digital: El Diario Vasco.
Autor: Teresa Flaño.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://www.diariovasco.com/deportes/mas-deportes/deporte-federado-vuelve-20201214203907-nt.html>
7. **Titular:** *El deporte asturiano reclama al Gobierno regional que rectifique.*
Fuente digital: El Comercio.
Autor: César Sánchez.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://www.elcomercio.es/deportes/mas-deportes/deporte-asturiano-reclama-20201214015928-ntvo.html>

8. **Titular:** *El Atalanta, una máquina de fútbol.*
Fuente digital: Diario Córdoba.
Autor: Agencias.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: https://www.diariocordoba.com/noticias/deportes/atalanta-maquina-futbol_1401862.html
9. **Titular:** *El deporte ya genera el 3,3 por ciento del PIB de España.*
Fuente digital: Marca.
Autor: Gerardo Riquelme.
Fecha de Publicación: 23 / Noviembre / 2020.
Url: <https://www.marca.com/otros-deportes/2020/11/23/5fbba80dca474148258b45de.html>
10. **Titular:** *Alerta en el PSG por la nueva lesión del Neymar.*
Fuente digital: El País.
Autor: Luis Villaescusa.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://elpais.com/deportes/2020-12-14/alerta-en-el-psg-por-la-nueva-lesion-del-neymar.html?outputType=amp>
11. **Titular:** *Coe defiende las protestas en los Juegos: "El deporte debe ser un catalizador".*
Fuente digital: El Mundo.
Autor: Andrés Aragón.
Fecha de Publicación: 11 / Diciembre / 2020.
Url: <https://www.elmundo.es/deportes/mas-deporte/2020/12/11/5fd3c9e2fdddf4d6d8b466b.html>
12. **Titular:** *Messi, el taconazo de Law y el derbi geoestratégico.*
Fuente digital: El País.
Autor: Walter Oppenheimer.
Fecha de Publicación: 13 / Diciembre / 2020.
Url: <https://elpais.com/deportes/2020-12-13/messi-el-taconazo-de-law-y-el-derbi-geoestrategico.html>
13. **Titular:** *El Barça se descose.*
Fuente digital: El País.
Autor: Juan I. Irigoyen.
Fecha de Publicación: 13 / Diciembre / 2020.
Url: <https://elpais.com/deportes/2020-12-12/el-barca-se-descose.html>
14. **Titular:** *La pandemia alumbra un atletismo de escaparate y récords.*
Fuente digital: El País.
Autor: Carlos Arribas.
Fecha de Publicación: 13 / Diciembre / 2020.
Url: <https://elpais.com/deportes/2020-12-13/la-pandemia-alumbra-un-atletismo-de-escaparate-y-records.html>
15. **Titular:** *El deporte en Sevilla: una buena práctica con mucho recorrido.*
Fuente digital: El Diario.
Autor: Andrés Moreno Mora.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://www.eldiario.es/andalucia/blogs/euronautas/deporte-sevilla-buena>

-practica-recorrido_132_6497179.html

16. **Titular:** *Andalucía recibe la bandera que la distingue como Región Europea del Deporte para 2021, que ve "un anhelo realizado".*
Fuente digital: La Vanguardia.
Autor: Agencias.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://www.lavanguardia.com/local/sevilla/20201214/6120453/andalucia-recibe-bandera-distingue-region-europea-deporte-2021-ve-anhelo-realizado.html>
17. **Titular:** *El dictado de Kroos*
Fuente digital: El País.
Autor: Lorenzo Calonge.
Fecha de Publicación: 13 / Diciembre / 2020.
Url: <https://elpais.com/deportes/2020-12-12/el-dictado-de-kroos.html>
18. **Titular:** *Alonso lidera el «rookie test» con un atracón de vueltas*
Fuente digital: ABC.
Autor: Javier Asprón.
Fecha de Publicación: 16 / Diciembre / 2020.
Url: https://www.abc.es/deportes/formula-1/abci-alonso-lidera-rookie-test-atracon-vueltas-202012151622_noticia.html
19. **Titular:** *CP Miralvalle: un colegio con mucha clase*
Fuente digital: El Salto Diario.
Autor: Alain Presentación.
Fecha de Publicación: 13 / Diciembre / 2020.
Url: <https://www.elsaltodiario.com/deportes/cp-miralvalle-un-colegio-con-mucha-clase>
20. **Titular:** *Fútbol, un deporte al borde de la saturación*
Fuente digital: ABC.
Autor: Rubén Cañizares.
Fecha de Publicación: 16 / Diciembre / 2020.
Url: https://www.abc.es/deportes/futbol/abci-futbol-deporte-nunca-descansa-202012150129_noticia.html
21. **Titular:** *Los deportes de invierno piden al Gobierno que las estaciones de esquí abran en Navidad*
Fuente digital: AS.
Autor: AS.
Fecha de Publicación: 02 / Diciembre / 2020.
Url: https://as.com/masdeporte/2020/12/02/polideportivo/1606905753_894084.html
22. **Titular:** *No hay quien pare a Bada Huesca que se codea con los mejores en la ASOBAL*
Fuente digital: Radio Huesca.
Autor: Luis Abadías.
Fecha de Publicación: 13 / Diciembre / 2020.
Url: <https://www.radiohuesca.com/deportes/no-hay-quien-pare-a-bada-huesca-13122020-147864.html>
23. **Titular:** *El deporte español también se escribe en femenino*

Fuente digital: 20 minutos.

Autor: María Carbajo.

Fecha de Publicación: 25 / Noviembre / 2020.

Url: <https://www.20minutos.es/deportes/noticia/4485108/0/deporte-espanol-se-escribe-en-femenino/>

24. **Titular:** *La Copa del Rey de baloncesto 2021 se jugará en el Palacio de los Deportes de Madrid*

Fuente digital: El Español.

Autor: Jose Nieto.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: https://www.elespanol.com/deportes/baloncesto/20201215/copa-rey-baloncesto-jugara-palacio-deportes-madrid/543696083_0.html

25. **Titular:** *Muere Vince Reffet, el hombre volador, tras un accidente en Dubai*

Fuente digital: El Mundo.

Autor: Francisco Carrión.

Fecha de Publicación: 17 / Noviembre / 2020.

Url: <https://www.elmundo.es/deportes/2020/11/17/5fb416e1fdddf6d988b46b9.html>

26. **Titular:** *Logroño Deporte aportará 250.000 euros para apoyar a los equipos de fútbol referentes*

Fuente digital: La Vanguardia.

Autor: Redacción.

Fecha de Publicación: 18 / Noviembre / 2020.

Url: <https://www.lavanguardia.com/local/la-rioja/20201118/49537135247/logrono-deporte-aportara-250000-euros-para-apoyar-a-los-equipos-de-futbol-referentes.html>

27. **Titular:** *Los deportes más antiguos de la historia*

Fuente digital: Estadio Deportivo.

Autor: R. S.

Fecha de Publicación: 03 / Noviembre / 2020.

Url: <https://www.estadiodeportivo.com/polideportivo/2020/11/03/deportes-antiguos-historia/303412.html>

28. **Titular:** *Coronavirus en el deporte: El CSD no logra un acuerdo para un protocolo común de vuelta del deporte no profesional*

Fuente digital: RTVE.

Autor: RTVE.

Fecha de Publicación: 08 / Septiembre / 2020.

Url: <https://www.rtve.es/deportes/20200908/fracaso-negociacion-protocolo-comun-deporte-no-profesional-coronavirus-csd/2041715.shtml>

29. **Titular:** *El deporte catalán denuncia a la Generalitat*

Fuente digital: La Vanguardia.

Autor: Xavier G. Luque.

Fecha de Publicación: 02 / Noviembre / 2020.

Url: <https://www.lavanguardia.com/deportes/20201102/49169445519/gerard-esteva-ufec-covid-19.html>

30. **Titular:** *Sólo la segunda ola pudo derrotar a la promesa de la natación de Navia*

Fuente digital: La Voz de Asturias.

Autor: Nel Oliveira.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: <https://www.lavozdeasturias.es/noticia/deportes/2020/12/15/solo-segunda-ola-pudo-derrotar-promesa-natacion-navia/00031608030879900324706.htm>

♥ DOCUMENTOS PARA EL GLOSARIO DE SALUD.

1. **Titular:** *¿Qué es la esclerosis lateral amiotrófica (ELA)?: Síntomas, causas y cómo tratar la enfermedad.*
Fuente digital: La Vanguardia.
Autor: Redacción.
Fecha de Publicación: 21 / Junio / 2019.
Url: <https://www.lavanguardia.com/vida/salud/enfermedades/20190621/463017004582/ela-esclerosis-lou-gehrig-discapacidad-stephen-hawking-dia-mundial-de-la-ela.html>
2. **Titular:** *Calendarios vacunales: así son los de Portugal, Italia, Reino Unido y España.*
Fuente digital: Gaceta Médica.
Autor: Mónica Gail.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://gacetamedica.com/politica/calendarios-vacunales-asi-son-los-de-portugal-italia-reino-unido-y-espana/>
3. **Titular:** *Cataluña registra el mayor aumento de plazas de FSE respecto a la última convocatoria.*
Fuente digital: Gaceta Médica.
Autor: Mario Ruiz.
Fecha de Publicación: 10 / Diciembre / 2020.
Url: <https://gacetamedica.com/politica/cataluna-registra-el-mayor-aumento-de-plazas-de-fse-respecto-a-la-ultima-convocatoria/>
4. **Titular:** *“Objetivos cumplidos” en CAR-T dos años después de su integración en España.*
Fuente digital: La Gaceta Médica.
Autor: Alberto Cornejo.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://gacetamedica.com/politica/objetivos-cumplidos-en-car-t-dos-anos-despues-de-su-integracion-en-espana/>
5. **Titular:** *La COVID-19 persistente ya tiene un código en la Clasificación Internacional de Enfermedades de la OMS.*
Fuente digital: La Gaceta Médica.
Autor: Daniela González.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://gacetamedica.com/politica/la-covid-19-persistente-ya-tiene-un-codigo-en-la-clasificacion-internacional-de-enfermedades-de-la-oms/>
6. **Titular:** *Tras la aparición del nuevo coronavirus, la Organización Mundial de la Salud convoca al comité de emergencias*
Fuente digital: Noticias ONU.
Autor: Naciones Unidas.
Fecha de Publicación: 20 / Enero / 2020.
Url: <https://news.un.org/es/story/2020/01/1468221>
7. **Titular:** *Regulación, EECC y precio y reembolso: retos para mejorar el acceso a oncología de*

precisión en Europa

Fuente digital: La Gaceta Médica.

Autor: Nieves Sebastián.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: <https://gacetamedica.com/investigacion/regulacion-eecc-y-precio-y-reem-bolso-retos-para-mejorar-el-acceso-a-oncologia-de-precision-en-europa/>

8. **Titular:** *María Casanova-Acebes: “Mi grupo en el CNIO tratará de evitar la cronicidad del cáncer”.*

Fuente digital: La Gaceta Médica.

Autor: Gaceta Médica.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: <https://gacetamedica.com/investigacion/maria-casanova-acebes-mi-grupo-en-el-cnio-tratará-de-evitar-la-cronicidad-del-cancer/>

9. **Titular:** *Nuevos datos de belantamab mafodotina para el tratamiento del MM refractario*

Fuente digital: La Gaceta Médica.

Autor: Sandra Pulido.

Fecha de Publicación: 11 / Diciembre / 2020.

Url: <https://gacetamedica.com/investigacion/nuevos-datos-de-belantamab-mafodotina-para-el-tratamiento-del-mm-refractario/>

10. **Titular:** *La Enfermedad de Creutzfeldt-Jakob: cuando la incidencia se equipara a la prevalencia.*

Fuente digital: La Gaceta Médica.

Autor: Sandra Pulido.

Fecha de Publicación: 11 / Diciembre / 2020.

Url: <https://gacetamedica.com/investigacion/la-enfermedad-de-creutzfeldt-jakob-cuando-la-incidencia-se-equipara-a-la-prevalencia/>

11. **Titular:** *ASH 2020 consolida el beneficio clínico de Yescarta en LBCG y se abre paso en LNH*

Fuente digital: La Gaceta Médica.

Autor: Carmen M. López.

Fecha de Publicación: 11 / Diciembre / 2020.

Url: <https://gacetamedica.com/investigacion/oncologia/ash/ash-2020-consolida-el-beneficio-clinico-de-yescarta-en-lbcg-y-se-abre-paso-en-lnh/>

12. **Titular:** *Del tratamiento sistémico al tópico, desafío en la psoriasis leve*

Fuente digital: La Gaceta Médica.

Autor: Daniela González.

Fecha de Publicación: 11 / Diciembre / 2020.

Url: <https://gacetamedica.com/investigacion/del-tratamiento-sistemico-al-topico-desafio-en-la-psoriasis-leve/>

13. **Titular:** *Saturan hospitales en Jalisco por COVID-19.*

Fuente digital: El Informador.

Autor: El Informador.

Fecha de Publicación: 13 / Diciembre / 2020.

Url: <https://www.informador.mx/Saturan-hospitales-en-Jalisco-por-COVID-19-1202012130001.html>

14. **Titular:** *Expertos advierten acerca de cuántos minutos diarios son suficientes para deprimirse al navegar en redes sociales.*

Fuente digital: Entrepreneur.
Autor: Entrepreneur Staff.
Fecha de Publicación: 12 / Diciembre / 2020.
Url: <https://www.entrepreneur.com/article/361561>

15. **Titular:** *Solares: todo lo nuevo para protegernos del sol*
Fuente digital: Marie Claire.
Autor: Fernando Gomez Dossena.
Fecha de Publicación: 12 / Diciembre / 2020.
Url: <https://marieclaire.perfil.com/noticias/belleza/lo-nuevo-en-proteccion-solar.phtml>
16. **Titular:** *Coronavirus: Qué cuidados hay que tener al vacunar a personas alérgicas.*
Fuente digital: Noticias Perfil.
Autor: Stella Maris Cuevas.
Fecha de Publicación: 11 / Diciembre / 2020.
Url: <https://noticias.perfil.com/noticias/opinion/que-cuidados-tener-al-vacunar-contr-el-coronavirus-a-personas-alergicas.phtml>
17. **Titular:** *“El estigma existente sobre salud mental hace la vida más difícil a los pacientes”.*
Fuente digital: Con Salud.
Autor: Marisol Díaz.
Fecha de Publicación: 10 / Octubre / 2019.
Url: https://www.consalud.es/pacientes/dias-mundiales/el-estigma-existente-sobre-salud-mental-hace-la-vida-mas-dificil-a-los-pacientes_69278_102.html
18. **Titular:** *La OMS declara emergencia de salud pública internacional el brote de ébola del Congo.*
Fuente digital: El País.
Autor: El País.
Fecha de Publicación: 17 / Julio / 2019.
Url: https://elpais.com/sociedad/2019/07/17/actualidad/1563386518_647191.html
19. **Titular:** *Salud amplía a los andaluces con diabetes el sistema flash de control de la glucosa.*
Fuente digital: ABC Andalucía.
Autor: S.A.
Fecha de Publicación: 13 / Noviembre / 2019.
Url: https://sevilla.abc.es/andalucia/sevi-salud-amplia-andaluces-diabetes-sistema-flash-control-glucosa-201911131205_noticia.html
20. **Titular:** *Delhi reparte millones de máscaras tras declarar la emergencia de salud pública por contaminación.*
Fuente digital: El País.
Autor: Ángel Martínez.
Fecha de Publicación: 01 / Noviembre / 2019.
Url: https://elpais.com/sociedad/2019/11/01/actualidad/1572637894_956255.html
21. **Titular:** *El ébola, a un paso de ser una emergencia de salud pública internacional.*
Fuente digital: Noticias ONU.
Autor: Naciones Unidas.
Fecha de Publicación: 15 / Julio / 2019.
Url: <https://news.un.org/es/story/2019/07/1459281>

22. **Titular:** *La crisis climática dañará toda su vida a un bebé que nazca hoy.*
Fuente digital: La Vanguardia.
Autor: Cristina Saez.
Fecha de Publicación: 14 / Noviembre / 2019.
Url: <https://www.lavanguardia.com/ciencia/20191114/471582229720/informe-lancet-cambio-climatico-amenaza-salud-ninos.html>
23. **Titular:** *La dieta perfecta para salvar el planeta y la salud del ser humano.*
Fuente digital: El País.
Autor: Manuel Planelles y Laura Delle Femmine.
Fecha de Publicación: 17 / Enero / 2019.
Url: https://elpais.com/sociedad/2019/01/16/actualidad/1547667687_190434.html
24. **Titular:** *Diabetes: bulos y mitos que te están intentado colar sobre esta enfermedad*
Fuente digital: Maldita: Malditobulo.
Autor: Maldito Bulo.
Fecha de Publicación: 14 / Noviembre / 2019.
Url: <https://maldita.es/malditobulo/2019/11/14/diabetes-bulos-y-mitos-que-te-estan-intentado-colar-sobre-esta-enfermedad/>
25. **Titular:** *Enfermedad de Crohn, el trastorno contra el que luchaba Sergio Gendler.*
Fuente digital: Infobae.
Autor: Valeria Chavez.
Fecha de Publicación: 13 / Junio / 2019.
Url: <https://www.infobae.com/salud/2019/06/13/cual-es-la-enfermedad-contra-la-que-sergio-gendler-lucho-con-hidalguia/>
26. **Titular:** *¿Qué es el Parkinson?: Síntomas, causas y cómo tratar la enfermedad.*
Fuente digital: La Vanguardia.
Autor: Redacción.
Fecha de Publicación: 11 / Abril / 2019.
Url: <https://www.lavanguardia.com/vida/salud/enfermedades/20190411/461586695549/parkinson-temblores-cuerpos-de-lewy-dopamina-neurodegenerativo-neuronas.html>
27. **Titular:** *¿Qué es el paludismo o malaria?: Síntomas, causas y cómo tratar la enfermedad.*
Fuente digital: La Vanguardia.
Autor: Redacción.
Fecha de Publicación: 25 / Abril / 2019.
Url: <https://www.lavanguardia.com/vida/salud/enfermedades-infecciosas/20190425/461854698508/paludismo-malaria-mosquito-fiebre-alta-anofeles-tropical-subsaariana.html>
28. **Titular:** *Alzheimer: nueve cosas que puedes hacer para intentar evitarlo.*
Fuente digital: La Vanguardia.
Autor: Mayte Rius.
Fecha de Publicación: 21 / Septiembre / 2019.
Url: <https://www.lavanguardia.com/vivo/lifestyle/20190921/47478613075/dia-mundial-alzheimer-como-prevenir-evitar.html>
29. **Titular:** *¿Qué es la hipertensión arterial?: Síntomas, causas y cómo tratar la enfermedad.*
Fuente digital: La Vanguardia.
Autor: Redacción.

Fecha de Publicación: 17 / Mayo / 2019.

Url: <https://www.lavanguardia.com/vida/salud/enfermedades-cardiovasculares/20190517/462290514298/hipertension-arterial-presion-arterial-sistolica-diastolica-corazon-sobreesfuerzo.html>

30. **Titular:** *¿Qué es la hemofilia?: Síntomas, causas y cómo tratar la enfermedad.*

Fuente digital: La Vanguardia.

Autor: Redacción.

Fecha de Publicación: 17 / Abril / 2019.

Url: <https://www.lavanguardia.com/vida/salud/enfermedades-geneticas/20190417/461706818788/hemofilia-sangre-hemorragia-coagulacion-cromosoma-x-heridas-factor-viii-factor-ix-sangrado.html>

♥ DOCUMENTOS PARA EL GLOSARIO DE POLÍTICA.

1. **Titular:** *Política de déficit cero y ayudas millonarias a los autónomos, así financia Alemania el cierre.*

Fuente digital: ABC.

Autor: Rosalía Sánchez.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: https://www.abc.es/economia/abci-politica-deficit-cero-y-ayudas-millonarias-autonomos-financia-alemania-cierre-202012140916_noticia_amp.html

2. **Titular:** *Otegi dice que el Supremo pretende "desestabilizar el escenario político".*

Fuente digital: El Diario.

Autor: El Diario.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: https://www.eldiario.es/politica/otegi-dice-supremo-pretende-desestabilizar-escenario-politico.1_6506366.html

3. **Titular:** *Los cuatro días de Errejón.*

Fuente digital: La Vanguardia.

Autor: Manuel Pedro Vallín.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: <https://www.lavanguardia.com/politica/20201214/6118225/cuatro-dias-erregon.html>

4. **Titular:** *La guerra y la política a la carta.*

Fuente digital: La Opinión de Málaga.

Autor: Ángel Vence.

Fecha de Publicación: 13 / Diciembre / 2020.

Url: <https://amp.laopiniondemalaga.es/opinion/2020/12/14/guerra-politica-carta/1211111.html>

5. **Titular:** *El negacionismo en política.*

Fuente digital: Excelsior.

Autor: Ricardo Pascoe Pierce.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: <https://www.excelsior.com.mx/opinion/ricardo-pascoe-pierce/el-negacionismo-en-politica/1422108>

6. **Titular:** *La política digital no es blanca o negra, sino gris.*

Fuente digital: El Español.

Autor: Elena Arrieta.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: https://www.elespanol.com/invertia/disruptores-innovadores/opinion/20201214/politica-digital-no-blanca-negra-gris/542825720_13.html

7. **Titular:** *Rodrigo Rato: Cuando la política se estropea, lo demás también*.

Fuente digital: 20 minutos.

Autor: 20 minutos.

Fecha de Publicación: 13 / Diciembre / 2020.

Url: <https://www.20minutos.es/noticia/4509727/0/rodrigo-rato-politica-estropea-demas-tambien/>

8. **Titular:** *Política Calígula*.

Fuente digital: El Periódico.

Autor: Gabriel Rufián.

Fecha de Publicación: 12 / Diciembre / 2020.

Url: <https://www.elperiodico.com/es/opinion/20201212/politica-caligula-11381907>

9. **Titular:** *Otorgar autoridad para anular el debate sobre política económica*.

Fuente digital: Naiz.

Autor: Isidro Esnaola.

Fecha de Publicación: 13 / Abril / 2020.

Url: <https://www.naiz.eus/eu/info/noticia/20201213/otorgar-autoridad-para-anular-el-debate-sobre-politica-economica>

10. **Titular:** *González Laya cree que la UE mantendrá el reconocimiento "político".^a Guaidó pero no aclara con qué título*.

Fuente digital: Europapress.

Autor: Europapress.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: <https://www.europapress.es/nacional/noticia-gonzalez-laya-cree-ue-mantendra-reconocimiento-politico-guaido-no-aclara-titulo-20201214110358.html>

11. **Titular:** *La dieta perfecta para salvar el planeta y la salud del ser humano*.

Fuente digital: The World News.

Autor: The World News.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: <https://theworldnews.net/es-news/los-sondeos-de-la-junta-revelan-que-todos-los-caminos-para-la-derecha-en-andalucia-pasan-por-vox>

12. **Titular:** *Los socios de Sánchez acusan al Supremo de «persecución política» cuando las causas no les favorecen*.

Fuente digital: Ok Diario.

Autor: Juanan Jiménez.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: <https://okdiario.com/espana/otro-socio-sanchez-acusa-tribunal-supremo-persecucion-politica-investigar-corrupcion-del-pnv-6575772>

13. **Titular:** *El PNV denuncia una "persecución política".^{en} la causa abierta por el Supremo al consejero navarro Manuel Ayerdi por el 'caso Davalor'*.

Fuente digital: El Diario.

Autor: El Diario Navarra.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: https://www.eldiario.es/navarra/pnv-denuncia-persecucion-politica-causa-abierta-supremo-consejero-navarro-manuel-ayerdi-caso-davalor_1_6505352.html

14. **Titular:** *Del Cid: “Torremolinos se ha convertido en reflejo de la peor política, la que se olvida de la gente”.*

Fuente digital: Revista Lugar de Encuentro.

Autor: Lugar de Encuentro.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: <https://www.revistalugardeencuentro.com/wp/2020/12/15/del-cid-torremolinos-se-ha-convertido-en-reflejo-de-la-peor-politica-la-que-se-olvida-de-la-gente/>

15. **Titular:** *C-LM destaca el papel de la Política de Cohesión para garantizar la igualdad entre las diferentes regiones de Europa.*

Fuente digital: La Vanguardia.

Autor: Agencias.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: <https://www.lavanguardia.com/local/castilla-la-mancha/20201215/6123426/c-lm-destaca-papel-politica-cohesion-garantizar-igualdad-diferentes-regiones-europa.html>

16. **Titular:** *Mazón reivindica una política hídrica que evite la “guerra de regiones”.*

Fuente digital: La Razón.

Autor: M.S.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: https://www.larazon.es/comunidad-valenciana/20201215/mktvbsfazvfvlaeiti_kh22uh6e.html

17. **Titular:** *Asturias pide «calibrar y dotar adecuadamente» la nueva «herramienta de política industrial».*

Fuente digital: El Comercio.

Autor: Noelia A. Erasquin.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: <https://www.elcomercio.es/economia/estatuto-electrointensivo-principado-rechaza-20201215171412-nt.html>

18. **Titular:** *Federalismo, participación política, convivencia y municipalismo, algunas lecturas y reflexiones.*

Fuente digital: AraInfo.

Autor: Pedro Santistevé.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: <https://arainfo.org/federalismo-participacion-politica-convivencia-y-municipalismo-algunas-lecturas-y-reflexiones/>

19. **Titular:** *Telemadrid pide “voluntad política.” al Gobierno regional para solucionar los problemas de futuro en la cadena.*

Fuente digital: VerTele.

Autor: EuropaPress Redacción.

Fecha de Publicación: 15 / Diciembre / 2020.

Url: https://vertele.eldiario.es/noticias/Telemadrid-voluntad-politica-Gobierno-Isabel-Diaz-Ayuso-solucionar-problemas-dialogo_0_2295970429.html

20. **Titular:** *Inmigración en la UE: el 2020 termina sin avances en política de asilo.*
Fuente digital: DW.
Autor: Bernd Riegert.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://www.dw.com/es/inmigraci%C3%B3n-en-la-ue-el-2020-termina-sin-avances-en-pol%C3%ADtica-de-asilo/a-55941241>
21. **Titular:** *Luisa Broto denuncia que “la Política Cultural de Zaragoza está en manos de Vox”.*
Fuente digital: AraInfo.
Autor: AraInfo Redacción.
Fecha de Publicación: 15 / Diciembre / 2020.
Url: <https://arainfo.org/luisa-broto-denuncia-que-la-politica-cultural-de-zaragoza-esta-en-manos-de-vox/>
22. **Titular:** *Montero receta “inteligencia política” a sus socios de Podemos tras los últimos choques.*
Fuente digital: La Razón.
Autor: Ainhoa Martínez.
Fecha de Publicación: 15 / Diciembre / 2020.
Url: <https://www.larazon.es/espana/20201215/njw2f4rqq5cznb64iqum4qq2y4.html>
23. **Titular:** *Iturgaiz afea al Gobierno vasco que solo «corteje» a Podemos, «su socio en Madrid», con los Presupuestos.*
Fuente digital: El Correo.
Autor: Lorena Gil.
Fecha de Publicación: 15 / Diciembre / 2020.
Url: <https://www.elcorreo.com/politica/euskadi-iturgaiz-pp-ciudadanos-politica-20201215110206-nt.html>
24. **Titular:** *Indignación en Bruselas ante la deriva en política económica de Pedro Sánchez.*
Fuente digital: Ok Diario.
Autor: Mario Moratalla.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://okdiario.com/economia/indignacion-bruselas-deriva-politica-economica-pedro-sanchez-6564077>
25. **Titular:** *Putin felicita finalmente a Biden por su victoria en las elecciones.*
Fuente digital: Expansión.
Autor: EFE.
Fecha de Publicación: 15 / Diciembre / 2020.
Url: <https://www.expansion.com/economia/politica/2020/12/15/5fd883c1468aeba2248b459e.html>
26. **Titular:** *¿Nueva política social?: Los impactos sociales de la pandemia dejarán profundas huellas. El diseño de la política social post-covid debe arrancar.*
Fuente digital: Portafolio.
Autor: Francisco Miranda Hamburger.
Fecha de Publicación: 14 / Diciembre / 2020.
Url: <https://www.portafolio.co/opinion/editorial/nueva-politica-social-editorial-francisco-miranda-547523>
27. **Titular:** *Una dirigencia política desubicada.*
Fuente digital: InfoBae.

Autor: Julio Bárbaro.

Fecha de Publicación: 13 / Diciembre / 2020.

Url: <https://www.infobae.com/opinion/2020/12/13/una-dirigencia-politica-desubi-cada/>

28. **Titular:** *Trump es la expresión de los nuevos tiempos que llegaron para quedarse en la política.*

Fuente digital: CNN.

Autor: Roberto Izurieta.

Fecha de Publicación: 11 / Diciembre / 2020.

Url: <https://cnnespanol.cnn.com/2020/12/11/opinion-trump-es-la-expresion-de-los-nuevos-tiempos-que-llegaron-para-quedarse-en-la-politica/>

29. **Titular:** *Albert Rivera: "Nunca he visto a nadie en política con el sentido de posesión del poder que tiene Pedro Sánchez".*

Fuente digital: Antena 3.

Autor: Espejo Público.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: <https://www.anten3.com/programas/espejo-publico/noticias/albert-rivera-presenta-libro-ciudadano-libre-menos-mal-que-dimiti.202012145fd7145e59ccfa00017eb02c.html>

30. **Titular:** *Santa María del Páramo aprueba el presupuesto y reclama .altura política.*

Fuente digital: Diario de León.

Autor: EFE.

Fecha de Publicación: 14 / Diciembre / 2020.

Url: <https://www.diariodeleon.es/articulo/provincia/santa-maria-paramo-aprueba-presupuesto-reclama-altura-politica/202012141847232069813.html>

B. Glosario de términos por tema

DEPORTES

♥ deporte
♥ deportivo
♥ jugar
♥ final
♥ grupo
♥ jugador
♥ equipar
♥ equipo
♥ rival
♥ club
♥ mundial
♥ champions
♥ marcar
♥ ganar
♥ gol
♥ técnico
♥ competición
♥ perder
♥ fútbol
♥ atletismo
♥ atleta
♥ deportista
♥ federación
♥ temporada
♥ entrenar
♥ entrenador
♥ derbi
♥ olímpico
♥ lesionar
♥ estadio

POLÍTICA

♥ político
♥ gobernar
♥ tribunal
♥ derecho
♥ social
♥ presidente
♥ país
♥ economía
♥ consejero
♥ ministro
♥ presupuesto
♥ decisión
♥ laboral
♥ electoral
♥ diputado
♥ voto
♥ jurídico
♥ autoridad
♥ izquierda
♥ derecha
♥ independentista
♥ socialista
♥ deuda
♥ crisis
♥ parlamentario
♥ parlamento
♥ portavoz
♥ fiscal
♥ judicial
♥ coalición

SALUD

♥ paciente
♥ enfermedad
♥ tratamiento
♥ vacuna
♥ salud
♥ célula
♥ terapia
♥ vacunación
♥ investigación
♥ cáncer
♥ covid-19
♥ clínico
♥ hospital
♥ síntoma
♥ síndrome
♥ inmunitario
♥ virus
♥ oncología
♥ cuidar
♥ cuidados
♥ estudio
♥ afectar
♥ contagio
♥ contagioso
♥ contagiar
♥ diagnóstico
♥ dosis
♥ oms
♥ fármaco
♥ patología

C. Selección a mano del glosario

■ Selección glosario deporte:

(‘haber’, 86), (‘deporte’, 52), (‘ser’, 48), (‘partir’, 30), (‘año’, 30), (‘tener’, 30), (‘estar’, 30), (‘deportivo’, 27), (‘jugar’, 26), (‘final’, 22), (‘grupo’, 21), (‘jugador’, 20), (‘hacer’, 20), (‘madrid’, 20), (‘equipar’, 20), (‘rival’, 19), (‘pasar’, 19), (‘vez’, 18), (‘club’, 18), (‘volver’, 18), (‘seguir’, 18), (‘después’, 18), (‘mundial’, 17), (‘champions’, 17), (‘marcar’, 17), (‘coe’, 17), (‘llegar’, 16), (‘octavo’, 16), (‘decir’, 16), (‘récord’, 16), (‘españa’, 15), (‘sevilla’, 15), (‘mundo’, 15), (‘contar’, 14), (‘primero’, 14), (‘dar’, 14), (‘ganar’, 13), (‘deber’, 13), (‘técnico’, 13), (‘competición’, 13), (‘city’, 13), (‘quedar’, 13), (‘les’, 13), (‘barça’, 12), (‘perder’, 12), (‘gran’, 12), (‘messi’, 12), (‘cambiar’, 12), (‘día’, 12), (‘generar’, 12), (‘koeman’, 12), (‘ciudad’, 12), (‘querer’, 11), (‘ahora’, 11), (‘equipo’, 11), (‘cómo’, 11), (‘último’, 11), (‘mejorar’, 11), (‘mejor’, 11), (‘gol’, 11), (‘actividad’, 11), (‘atletismo’, 11), (‘millón’, 11), (‘vida’, 10), (‘internacional’, 10), (‘salir’, 10), (‘atlético’, 10), (‘real’, 10), (‘además’, 10), (‘semana’, 10), (‘nuevo’, 10), (‘deportista’, 10), (‘zapatilla’, 10), (‘rugby’, 9), (‘levantar’, 9), (‘federación’, 9), (‘grande’, 9), (‘hablar’, 9), (‘league’, 9), (‘temporada’, 9), (‘cruzar’, 9), (‘psg’, 9), (‘encontrar’, 9), (‘barcelona’, 9), (‘momento’, 9), (‘nivel’, 9), (‘poder’, 9), (‘derecho’, 9), (‘tres’, 9), (‘falto’, 9), (‘azulgrana’, 9), (‘manera’, 9), (‘pandemia’, 9), (‘united’, 9), (‘maratón’, 9), (‘único’, 8), (‘recordar’, 8), (‘cuatro’, 8), (‘carrera’, 8), (‘ir’, 8), (‘dejar’, 8), (‘buscar’, 8), (‘equilibrio’, 8), (‘conocer’, 8), (‘fuerte’, 8), (‘sentir’, 8), (‘casar’, 8), (‘llevar’, 8), (‘caer’, 8), (‘atalantar’, 8), (‘entrenador’, 8), (‘error’, 8), (‘mantener’, 8), (‘aun’, 8), (‘ayudar’, 8), (‘durante’, 8), (‘fútbol’, 8), (‘convertir’, 8), (‘turismo’, 8), (‘social’, 8), (‘acabar’, 7), (‘explicar’, 7), (‘incluir’, 7), (‘importante’, 7), (‘español’, 7), (‘atacar’, 7), (‘sortear’, 7), (‘alemán’, 7), (‘español’, 7), (‘neymar’, 7), (‘olímpico’, 7), (‘coser’, 7), (‘creer’, 7), (‘juego’, 7), (‘embargar’, 7), (‘visitar’, 7), (‘mesar’, 7), (‘defender’, 7), (‘griezmann’, 7), (‘presidente’, 7), (‘impactar’, 7), (‘sector’, 7), (‘económico’, 7), (‘lesionar’, 7), (‘atleta’, 7), (‘valencia’, 7), (‘estadio’, 7), (‘entrenar’, 6), (‘derbi’, 6)

■ Selección glosario política:

(‘político’, 78), (‘haber’, 73), (‘ser’, 51), (‘estar’, 34), (‘económico’, 31), (‘gobernar’, 30), (‘partir’, 29), (‘decir’, 27), (‘supremo’, 25), (‘hacer’, 24), (‘casar’, 24), (‘tribunal’, 24), (‘año’, 23), (‘empresa’, 20), (‘euro’, 20), (‘debatir’, 19), (‘pasar’, 18), (‘millón’, 17), (‘derecho’, 17), (‘país’, 17), (‘punto’, 17), (‘general’, 16), (‘público’, 16), (‘navarro’, 16), (‘social’, 16), (‘presidente’, 15), (‘creer’, 15), (‘esquerra’, 15), (‘españa’, 14), (‘causar’, 14), (‘europeo’, 14), (‘economía’, 14), (‘vez’, 13), (‘ayudar’, 13), (‘durante’, 12), (‘deber’, 12), (‘presentar’, 12), (‘querer’, 12), (‘comunicación’, 12), (‘asunto’, 12), (‘dar’, 12), (‘centrar’, 11), (‘poder’, 11), (‘ver’, 11), (‘además’, 11), (‘formar’, 11), (‘ciudadano’, 11), (‘premiar’, 11), (‘europa’, 11), (‘consejero’, 11), (‘ministro’, 10), (‘presupuesto’, 10), (‘ir’, 10), (‘jornada’, 10), (‘personar’, 10), (‘recibir’, 10), (‘nuevo’, 10), (‘poner’, 10), (‘único’, 10), (‘tres’, 10), (‘ejemplo’, 10), (‘decisión’, 10), (‘cuestionar’, 10), (‘ahora’, 10), (‘medio’, 10), (‘laboral’, 10), (‘hoy’, 10), (‘hablar’, 10), (‘imponer’, 10), (‘psoe’, 10), (‘pnv’, 10), (‘ayerdi’, 10), (‘mantener’, 9), (‘pedir’, 9), (‘contar’, 9), (‘bildu’, 9), (‘cómo’, 9), (‘vox’, 9), (‘todavía’, 9), (‘líder’, 9), (‘quien’, 9), (‘mejor’, 9), (‘diputar’, 9), (‘llevar’, 9), (‘seguir’, 9), (‘realidad’, 9), (‘situación’, 9), (‘trump’, 9), (‘electoral’, 9), (‘voto’, 9), (‘resultar’, 9), (‘economista’, 9), (‘autoridad’, 9), (‘pandemia’, 8), (‘hora’, 8), (‘día’, 8), (‘suponer’, 8), (‘mesar’, 8), (‘otegi’, 8), (‘repetir’, 8), (‘jurídico’, 8), (‘izquierdo’, 8), (‘independentista’, 8), (‘abrir’, 8), (‘tener’, 8), (‘mundo’, 8), (‘unión’, 8), (‘preguntar’, 8), (‘adelantar’, 8), (‘alto’, 8), (‘nobel’, 8), (‘destacar’, 8), (‘socialista’, 8), (‘afectar’, 7), (‘deuda’, 7), (‘primero’, 7), (‘banco’, 7), (‘diferente’, 7), (‘mayor’, 7), (‘fondo’, 7), (‘perder’, 7), (‘crisis’, 7), (‘recordar’, 7), (‘actual’, 7), (‘parlamentar’, 7), (‘formación’, 7), (‘portavoz’, 7), (‘esperar’, 7), (‘parecer’, 7), (‘explicar’, 7), (‘futuro’, 7), (‘decretar’, 7), (‘tratar’, 7), (‘rato’, 7), (‘ciencia’, 7), (‘desarrollar’, 7), (‘bendodo’, 7), (‘persecución’, 7), (‘hualde’, 7), (‘torremolinos’, 7), (‘venir’, 6), (‘cuatro’, 6), (‘fraude’, 6), (‘inversión’, 6), (‘existir’, 6), (‘fiscal’, 6), (‘utilizar’, 6), (‘préstamo’, 6), (‘pagar’, 6), (‘aumentar’, 6), (‘juicio’, 6), (‘coalición’, 6)

■ Selección glosario Salud:

(‘paciente’, 64), (‘haber’, 63), (‘enfermedad’, 62), (‘ser’, 53), (‘tratamiento’, 50), (‘año’, 43), (‘vacuna’, 33), (‘ciento’, 33), (‘salud’, 32), (‘sistema’, 31), (‘nuevo’, 30), (‘caso’, 30), (‘estar’, 30), (‘célula’, 27), (‘primero’, 25), (‘terapia’, 25), (‘desarrollar’, 24), (‘car-t’, 24), (‘psoriasis’, 24), (‘vacunación’, 22), (‘tratar’, 21), (‘investigación’, 21), (‘plan’, 21), (‘país’, 21), (‘personar’, 21), (‘cáncer’, 21), (‘mayor’, 21), (‘día’, 20), (‘resultar’, 20), (‘covid-19’, 20), (‘experto’, 20), (‘pasar’, 19), (‘clínico’, 19), (‘respuesta’, 19), (‘hospital’, 19), (‘avanzar’, 18), (‘mesar’, 18), (‘indicar’, 18), (‘partir’, 17), (‘médico’, 17), (‘durante’, 17), (‘solicitud’, 17), (‘medir’, 16), (‘dato’, 16), (‘incluir’, 16), (‘mejorar’, 16), (‘calendario’, 16), (‘nacional’, 16), (‘estudiar’, 16), (‘plaza’, 15), (‘causar’, 14), (‘españa’, 14), (‘inmunitario’, 14), (‘respetar’, 14), (‘virus’, 14), (‘llegar’, 14), (‘tener’, 14), (‘explicar’, 14), (‘linfoma’, 14), (‘deber’, 14), (‘social’, 14), (‘hacer’, 13), (‘poder’, 13), (‘realizar’, 13), (‘tres’, 13), (‘destacar’, 13), (‘administrar’, 13), (‘grupo’, 13), (‘contar’, 13), (‘oncología’, 13), (‘confirmar’, 12), (‘después’, 12), (‘portugal’, 12), (‘cuatro’, 12), (‘niño’, 12), (‘llevar’, 12), (‘formar’, 12), (‘especialidad’, 12), (‘director’, 12), (‘coronavirus’, 12), (‘ecj’, 12), (‘cuidar’, 12), (‘solar’, 12), (‘ela’, 11), (‘estudio’, 11), (‘existir’, 11), (‘afectar’, 11), (‘general’, 11), (‘último’, 11), (‘suponer’, 11), (‘precisión’, 11), (‘centrar’, 11), (‘dar’, 11), (‘contagiar’, 11), (‘refractario’, 11), (‘provocar’, 10), (‘fase’, 10), (‘diagnóstico’, 10), (‘además’, 10), (‘italia’, 10), (‘unir’, 10), (‘acceso’, 10), (‘tipo’, 10), (‘decir’, 10), (‘mejor’, 10), (‘medicinar’, 10), (‘analizar’, 10), (‘importante’, 10), (‘redar’, 10), (‘protección’, 10), (‘esclerosis’, 9), (‘lateral’, 9), (‘amiotrófica’, 9), (‘síntoma’, 9), (‘señalar’, 9), (‘principal’, 9), (‘aspecto’, 9), (‘vida’, 9), (‘reinar’, 9), (‘vacunal’, 9), (‘europeo’, 9), (‘dosis’, 9), (‘semana’, 9), (‘riesgo’, 9), (‘número’, 9), (‘seguir’, 9), (‘poner’, 9), (‘grande’, 9), (‘efecto’, 9), (‘síndrome’, 9), (‘sentir’, 9), (‘área’, 9), (‘continuar’, 9), (‘internacional’, 9), (‘oms’, 9), (‘casar’, 9), (‘fármaco’, 9), (‘puig’, 9), (‘mil’, 9), (‘cómo’, 8), (‘dificultar’, 8), (‘problema’, 8), (‘reducir’, 8), (‘estrategia’, 8), (‘patología’, 8), (‘contagio’, 4)

D. Stop-words de spacy

{ 'tan', 'realizó', 'intentan', 'sido', 'ni', 'cuanta', 'parte', 'unos', 'incluso', 'alguno', 'uso', 'hacemos', 'cons
eguir', 'enfrente', 'mucha', 'parece', 'sean', 'buena', 'luego', 'vaya', 'hago', 'sin', 'más', 'allí', 'última', 'la
rgo', 'al', 'general', 'tenido', 'después', 'raras', 'través', 'uno', 'con', 'realizado', 'trabajo', 'podría', 'aque
llo', 'comentó', 'para', 'indicó', 'cierta', 'estado', 'menos', 'esas', 'hizo', 'podría', 'hasta', 'solo', 'dío', 'f
uimos', 'adenás', 'informo', 'así', 'nadie', 'la', 'ser', 'verdadera', 'últimos', 'su', 'aquellos', 'estas', 'día',
'podrá', 'están', 'vuestro', 'del', 'primera', 'entonces', 'podríamos', 'arriba', 'quizá', 'quedó', 'tampoco', 'ti',
'toda', 'un', 'podrías', 'verdad', 'esa', 'sobre', 'habla', 'quiere', 'soy', 'diferente', 'estará', 'días', 'horas',
'mencionó', 'quizás', 'fue', 'nuevas', 'hacerlo', 'empleamos', 'estas', 'ni', 'son', 'estaba', 'usa', 'este', 'mio
s', 'mía', 'ir', 'nuestras', 'ningunos', 'posible', 'breve', 'dieron', 'vosotras', 'después', 'tenía', 'nunca', 'ell
a', 'empleas', 'final', 'país', 'trabaja', 'creo', 'tarde', 'yo', 'le', 'haber', 'vamos', 'usas', 'anterior', 'estan
' 'espacio', 'estamos', 'debajo', 'existe', 'último', 'él', 'cual', 'no', 'informó', 'mío', 'usan', 'lugar', 'nues
tros', 'dejó', 'usamos', 'total', 'ahora', 'usais', 'esos', 'sola', 'trabajas', 'algún', 'consigues', 'mayor', 'mías
' 'repente', 'tiempo', 'ningún', 'eranos', 'solos', 'suyo', 'conseguimos', 'acuerdo', 'bastante', 'hicieron', 'nues
tro', 'aquellos', 'cómo', 'esto', 'aun', 'apenas', 'empleo', 'manera', 'dar', 'también', 'nueva', 'nuestra', 'tengo',
' 'hacia', 'atrás', 'manifestó', 'vuestras', 'nos', 'durante', 'puedo', 'diferentes', 'ver', 'como', 'ese', 'ayer',
'mal', 'aquellas', 'cuándo', 'ninguna', 'les', 'otras', 'respecto', 'tienen', 'realizar', 'supuesto', 'saben', 'voso
tros', 'aquí', 'éste', 'tras', 'expresó', 'mucho', 'trabajan', 'que', 'será', 'tal', 'tuyas', 'mis', 'quienes', 'ésa
s', 'alrededor', 'poca', 'intentamos', 'cuanto', 'pueda', 'sí', 'suyas', 'da', 'dijo', 'seis', 'demás', 'henos', 'he
' 'así', 'adrede', 'contigo', 'alguna', 'contra', 'stendo', 'solo', 'cuál', 'empleais', 'ante', 'nada', 'solamente',
'el', 'había', 'propia', 'os', 'ésta', 'otra', 'dos', 'aproxinadamente', 'intentas', 'suya', 'verdadero', 'todavía
' 'tercera', 'van', 'dentro', 'fuera', 'quiza', 'haciendo', 'que', 'hacen', 'algo', 'cosas', 'aquella', 'podeis', 'sabeis',
'afirmó', 'muchos', 'ciertas', 'estuvo', 'grandes', 'ha', 'hecho', 'sabe', 'trabajais', 'mediante', 'igual',
'cada', 'porque', 'trabajamos', 'los', 'varios', 'hacer', 'eres', 'cuenta', 'cuánta', 'podrían', 'arribaabajo', 'e
sos', 'próximo', 'pasada', 'llegó', 'enseguida', 'segundo', 'todos', 'pesar', 'tu', 'ése', 'cuánto', 'principalmente
' 'míos', 'país', 'hacéis', 'habían', 'tanto', 'propios', 'varias', 'antano', 'hace', 'cuatro', 'aquí', 'junto', 's
ino', 'primeros', 'pronto', 'pues', 'otro', 'todavía', 'tendrán', 'las', 'tuvo', 'trata', 'siguiente', 'ya', 'mientr
as', 'quienes', 'mismos', 'sabes', 'modo', 'conocer', 'hoy', 'hubo', 'demasiado', 'encuentra', 'pero', 'adenas', 'dí
ce', 'último', 'aquél', 'emplear', 'añadió', 'hay', 'siete', 'antano', 'sería', 'propio', 'dado', 'ésa', 'misno', 'm
ias', 'claro', 'cuales', 'lleva', 'poner', 'tener', 'aseguró', 'poder', 'encina', 'propias', 'nosotros', 'existen',
'va', 'poco', 'momento', 'eran', 'ahí', 'pueden', 'dicho', 'saber', 'ambos', 'partir', 'cierto', 'gran', 'señaló',
'desde', 'qué', 'donde', 'tenga', 'cuántas', 'mas', 'cerca', 'ellos', 'fui', 'día', 'ello', 'cinco', 'estais', 'lado',
'allí', 'en', 'voy', 'vuestra', 'pasado', 'excepto', 'consigo', 'ciertos', 'cuántos', 'intentais', 'emplean', 'mi',
'deben', 'nuevo', 'dónde', 'ellas', 'segun', 'siempre', 'ningunas', 'sea', 'trabajar', 'buenas', 'nuevos', 'embarg
o', 'salvo', 'usar', 'decir', 'serán', 'había', 'ustedes', 'teneis', 'estaban', 'nosotras', 'gueno', 'se', 'todo',
'debido', 'temprano', 'quién', 'vais', 'tendrá', 'intentar', 'una', 'vuestros', 'sera', 'tres', 'veces', 'agregó', 'h
an', 'cualquier', 'segunda', 'querenos', 'fin', 'solas', 'haces', 'algunas', 'bien', 'próximos', 'llevar', 'cuantos',
'ninguno', 'misma', 'también', 'delante', 'aquella', 'podriais', 'peor', 'consiguen', 'éstos', 'entre', 'sigue',
'está', 'ahí', 'por', 'primer', 'muy', 'estos', 'puede', 'consideró', 'fueron', 'detrás', 'mismas', 'valor', 'consigu
e', 'dicen', 'quizas', 'haya', 'pocas', 'ex', 'medio', 'me', 'cuáles', 'mejor', 'pudo', 'debe', 'tú', 'próximo', 'tu
ya', 'dijeron', 'vez', 'eras', 'adelante', 'ejemplo', 'sus', 'ocho', 'aquellas', 'de', 'sabemos', 'era', 'sonos', 's
i', 'buen', 'cuantas', 'tuyos', 'podrán', 'estar', 'te', 'lo', 'algunos', 'tenenos', 'usted', 'últimas', 'nío', 'aqu
el', 'es', 'cuando', 'todas', 'pocas', 'tiene', 'estados', 'otros', 'bueno', 'considera', 'sois', 'primero', 'intent
a', 'bajo', 'unas', 'eso', 'según', 'deprisa', 'aunque', 'esta', 'antes', 'dan', 'podrían', 'habrá', 'estoy', 'quien
' 'muchas', 'soyos', 'intento', 'cast', 'días', 'buenos', 'mía', 'tus', 'actualmente', 'explicó', 'tuyo', 'hablan',
'menudo', 'detrás', 'lejos', 'sé', 'contigo', 'podenos', 'aún' }

E. Scripts utilizados

Código Dependencias.py

```
# Autores: Ana Maria Casado y Ana Sanmartin
#
# Este script instala en el sistema todas las dependencias necesarias para poder
# ejecutar los programas

# Paquetes necesarios para el funcionamiento del programa
import os

#Lista de paquetes a instalar
dependencias = ['pandas', 'spacy', 'gensim', 'nltk', 'sklearn', 'matplotlib', 'seaborn']

#Instalacion del comando pip3
os.system("sudo apt update")
os.system("sudo install python3-pip")

for s in dependencias:
    os.system("pip3 install " + s)
```

Código lector.py

```
# Autores: Ana Maria Casado y Ana Sanmartin
#
# Este script recoge datos de la linea de comandos introducido por el usuario.

# Paquetes necesarios para el funcionamiento del programa
import os
import argparse
import pandas as pd
from collections import Counter
from crear_corpus import pre_procesar_texto

## ESTABLECIDOS POR EL USUARIO ##
n = 15
out = os.getcwd()
glosario = 0
modo = 0

def main(directorio, n, modo):

    doc = ""
    files = ""
    temas = ["Deportes", "Politica", "Salud"]
    path = ""
    x = 'frecuencia'

    # En esta funcion lo que se realizara es obtener los nombres de los documentos
    # presentes en las carpetas
    # llamadas 'Deporte', 'Politica', 'Salud'.

    for i in temas:
        doc = ""
        path = directorio + "/Documentos/" + i + "/"
        for j in range(n):
            f = open(path + i.lower() + str(j+1) + ".txt", "r")
            files = f.read()
            doc = doc + files

        clean = pre_procesar_texto(doc.lower(), get_modo(modo))
        if(x == 'frecuencia'):
            word_freq = Counter(clean)
            common_words = word_freq.most_common()

        if(x == 'tfidf'):
            pass

    # Albergar en un documento los textos correspondientes a cada glosario.
    f2 = open(directorio + "/Pre-Glosario/" + i + ".txt", "w+")
    f2.write(str(common_words))
    f2.close()
```

```

# Funcion que determina el tipo de preprocesamiento del texto
def get_mod0(mod0):
    switcher = {
        0 : 'lema',
        1 : 'stem',
        2 : 'none'}
    return switcher.get(mod0)

#####
# PROGRAMA PRINCIPAL #
#####

# Argparse - Parametros a ser introducidos por el usuario
parser = argparse.ArgumentParser(description="Search by terms")

parser.add_argument('-d',
                    "--directorio",
                    type=str,
                    help="Directorio general donde se encuentran las carpetas con los
                    textos a procesar. El path debe de ser desde la raiz hasta la carpeta Documentos. Ej
                    : ../Document_classification/Documentos")

parser.add_argument('-n',
                    "--n",
                    type=int,
                    help="Numero de documentos a procesar para el glosario")

parser.add_argument('-m',
                    "--modo",
                    type=int,
                    help="Aplicacion de lemmalization, stemmer o nada. Para aplicarlo
                    introducir el numero correspondiente: 0 = lemmalization, 1 = stemmer, 2 = no aplicar
                    nada")

# Parseo de los argumentos
arguments = vars(parser.parse_args())

if arguments['directorio']:
    directorio = arguments['directorio']
else:
    print("ERROR: Porfavor introduzca palabras validas")
    exit()
if arguments['n']:
    if arguments['n'] > 0:
        n = arguments['n']
    else:
        print("ERROR: El valor de N debe ser mayor que 0")
        exit()
if arguments['modo']:
    if arguments['modo'] > 0 and arguments['modo'] < 3:
        modo = arguments['modo']
    else:
        print("ERROR: Introduzca un valor valido entre 0 y 2")
        exit()

main(directorio, n, modo)

```

Código crear_corpus.py

```

# Autores: Ana Maria Casado y Ana Sanmartin
#
# Este script de realizar el preprocesamiento de los textos tokenizando y limpiando el
# texto.

# Paquetes necesarios para el funcionamiento del programa
import spacy
import nltk
from nltk import SnowballStemmer
import os
import pandas as pd

```

```

def pre_procesar_texto(texto: str, mode: str):

    path = os.getcwd()
    stop_words = []

    #Eliminacion de los stopwords
    #nlp = spacy.load('es_core_news_sm') # Se decidio no utilizar las stopwords de spacy
    #porque nos parecian incompletas
    f = open(path + "/stop_words.txt", "r") #Usamos nuestras propias stopwords

    sw = f.readlines()

    # Strips the newline character
    for line in sw:
        stop_words.append(line.strip())

    #Tokenizacion y limpieza del texto

    doc = nlp(texto) # Crea un objeto de spacy tipo nlp
    tokens = [token for token in doc if not token.is_punct and len(token) > 2] #
    #Convierte las palabras en tokens
    clean = [token for token in tokens if not str(token) in stop_words and not
    isinstance(token, int)] #Elimina las stopwords del texto

    if(mode == 'lema'):
        clean = [token.lemma_.lower() for token in clean]

    elif(mode == 'stem'):
        pass
        spanishstemmer = SnowballStemmer('spanish')
        clean = [spanishstemmer.stem(token) for token in clean]
    return clean

```

Código clasificador.py

```

# Autores: Ana Maria Casado y Ana Sanmartin
#
# Este script se encarga recoger el directorio de trabajo, el numero de documentos a
# procesar,
# y generar el modelo que se utilizara para la clasificacion.

# Paquetes necesarios para el funcionamiento del programa
import os
import argparse
import pandas as pd
import numpy as np
import itertools
from operator import itemgetter
from crear_corpus import pre_procesar_texto
from gensim import corpora, models, similarities
from gensim.corpora.mmcorpus import MmCorpus
from nltk.tokenize import wordpunct_tokenize
from nltk.stem import PorterStemmer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, precision_score, recall_score
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction import text
from sklearn.metrics.pairwise import cosine_similarity
from nltk.tokenize.treebank import TreebankWordDetokenizer
from evaluator import get_evaluation

## Variables globales ##
directorio = ""
nmin = 16
rango = 15
modelo = 0

STOP_WORDS = []
temas = ["Deportes", "Politica", "Salud"]

def clasificador_documentos(directorio, n_min, rango, modelo):
    """
    Esta funcion recibe los datos que ha introducido el usuario y llama a otros metodos

```

```

para la creacion de los modelos

directorio = directorio de trabajo, path desde la raiz hasta .../
Documentation_classification
n_min = numero del documento mas bajo a ser analizado
rango = numero de documentos a ser analizados
modelo = la clase de modelo que se utilizara en el procesamiento
"""

#Carga de las stop_words
path = ""
f = open(directorio + "/stop_words.txt", "r")
sw = f.readlines()

for line in sw:
    STOP_WORDS.append(line.strip())

path_results = directorio + "/Resultados/"
path_glosario = directorio + "/Glosario/"

doc = []
doc_id = []

#Para cada tema, se elige cada documento y se almacena todo el texto en una lista
#para poder procesarlos conjuntamente
for i in temas:
    path = directorio + "/Documentos/" + i + "/"
    for j in range(n_min, n_min + rango):
        f = open(path + i.lower() + str(j) + ".txt", "r")
        files = f.read()
        doc += [files]
        doc_id += [i.lower() + str(j)]

#Si el modelo es tfidf, se creara un Bag of ords (BoW)
if(modelo == 0):
    dictionary = create_dictionary(path_glosario)
    #Pre-procesamiento de los documentos de test para generar un BoW
    bow = process_text(doc, dictionary)
else:
    #Si se trata de otro modelo, se pasara el documento completo sin procesar
    dictionary = None
    bow = doc

# Lanzar_clasificador llamara a la funcion del clasificador correspondiente
lanzar_clasificador(bow, doc_id, dictionary, path_glosario, path_results, modelo)

#####
# ALMACENAMIENTO DE RESULTADOS #
#####

def guardar_resultados(ranking, doc_id, path_results):
    """
    Esta funcion recibe los resultados de los modelos, los nombres de los documentos y
    un path donde almacenarlos. Se encarga de almacenar las matrices de salida de los
    resultados

    ranking = Matriz con las predicciones de pertenencia de documentos a una clase
    doc_id = nombre del documento almacenado
    path_id = directorio donde se almacenaran los documentos de resultados
    """

    res = open(path_results, "w")
    #Para cada fila en la matriz de ranking, escribir una linea en el txt de resultados
    for doc, score in enumerate(ranking):
        doc_string = doc_id[doc] + ' ' + str(score) + '\n'
        res.write(doc_string)
    res.close()

```

```

def guardar_clasificacion(dataframe, doc_id, path_clasificacion):
    """
    Esta funcion recibe las clasificaciones finales realizadas por los modelos, los
    nombres de los documentos
    y el path donde almacenarlos. Se encarga de almacenar las matrices de salida de las
    clasificaciones.

    dataframe = Matriz con las predicciones finales clasificadas de pertenencia de
    documentos a una clase
    doc_id = nombre del documento almacenado
    path_clasificacion = directorio donde se almacenaran los documentos de clasificacion
    """
    np.savetxt(path_clasificacion, dataframe.values, fmt='%s')

def guardar_evaluacion(dataframe, path_evaluacion):
    """
    Esta funcion recibe las clasificaciones finales realizadas por los modelos y el path
    donde se almacenara
    los valores para evaluar la bondad de los modelos. Se encarga de llamar a otro
    script.

    dataframe = Matriz con las predicciones finales clasificadas de pertenencia de
    documentos a una clase
    doc_id = nombre del documento almacenado
    path_evaluacion = directorio donde se almacenaran los documentos de evaluacion
    """

    get_evaluation(dataframe, path_evaluacion, temas)

#####
# MODELOS A UTILIZAR EN EL PROYECTO #
#####

#Dependiendo del valor de la variable modelo, la funcion lanzar_clasificador utilizara
la llamada al proceso correspondiente
def lanzar_clasificador(bow, doc_id, dictionary, path_glosario, path_results, m):
    """
    Esta funcion recibe el documento o el BoW, el diccionario, el path donde se
    encuentran almacenados los glosarios
    el path donde se almacenaran los resultados y el tipo de modelo a crear. Se encarga
    de llamar al modelo correspondiente.

    bow = Textos a ser analizados
    doc_id = nombre del documento almacenado
    dictionary = diccionario que relaciona vectores con las palabras
    path_glosario = carpeta donde se encuentran los glosarios almacenados
    path_results = carpeta donde se almacenaran los resultados
    m = tipo de modelo a utilizar
    """

    if(m == 0):
        tfidf_model(bow, doc_id, dictionary, path_glosario, path_results)
    elif(m == 1):
        word2vec_model(bow, doc_id, path_glosario, path_results)
    elif(m == 2):
        naivebayes_model(bow, doc_id, path_glosario, path_results)

def tfidf_model(bow, doc_id, dictionary, path_glosario, path_results):
    """
    Esta funcion recibe el documento o el BoW, los nombres de los documentos, el
    diccionario, el path donde se encuentran almacenados los glosarios
    el path donde se almacenaran los resultados. Se encarga de generar y aplicar el
    modelo doc2bow con tfidf.

    bow = Textos a ser analizados
    doc_id = nombre del documento almacenado
    dictionary = diccionario que relaciona vectores con las palabras
    path_glosario = carpeta donde se encuentran los glosarios almacenados

```

```

path_results = carpeta donde se almacenaran los resultados
"""

#Crea el modelo tfidf
tfidf = models.TfidfModel(bow)
index = similarities.SparseMatrixSimilarity(bow, num_features=len(dictionary))

final = pd.DataFrame()
max_values = pd.DataFrame()

#Para cada glosario almacenar los resultados en la carpeta tfidf
for filename in os.listdir(path_glosario):
    f2 = open(path_glosario + filename, "r")
    glosario = f2.read()
    clean_glosario = wordpunct_tokenize(glosario)

    #se aplica el modelo doc2bow
    tfidf_glosario = tfidf[dictionary.doc2bow(clean_glosario)]
    sims = index[tfidf_glosario]

    path_res_glosario = path_results+ "/tfidf/tfidf_" + filename
    guardar_resultados(sims, doc_id, path_res_glosario)
    final[cambiar_singlelabel(filename[:-13])] = sims.tolist()

#Creacion de una estructura tipo dataframe para estructurar los resultados
#La matriz contiene el nombre del documento, la clasificacion real, la probabilidad
#de prediccion
#y la clasificacion del modelo.
max_values['Documento'] = doc_id
max_values['Real'] = cambiar_label([i[:-2] for i in doc_id])
max_values['Valor'] = final.max(axis = 1)
max_values['Predicciones'] = final.idxmax(axis=1)

#Se guardan los resultados en clasificacion y en evaluacion
guardar_clasificacion(max_values, doc_id, path_results + "/Clasificacion/tfidf.txt")
guardar_evaluacion(max_values, path_results + "/Evaluacion/tfidf_eval.txt")

def get_embeddings_from_document(model, g):
    """
    Esta funcion recibe el modelo y una lista de palabras. Se encarga de calcular la
    media de los embeddings que existan
    tanto en el modelo como en la lista de palabras

    model = modelo creado
    g = lista de palabras en el documento
    """

    embeddings = []

    #para cada palabra de la lista, si la palabra esta presente en el modelo annadirla a
    la lista embeddings
    for word in g:
        #print(g)
        if word in model.wv:
            embeddings.append(model.wv[word])
            #print(model.wv)

    #calculo de la media
    mean = np.mean(embeddings, axis = 0) if embeddings != [] else np.zeros(10)
    return mean

def word2vec_model(bow, doc_id, path_glosario, path_results):
    """
    Esta funcion recibe el documento o el BoW, los nombres de los documentos, el path
    donde se encuentran almacenados los glosarios
    el path donde se almacenaran los resultados. Se encarga de generar y aplicar el
    modelo word2vec.

    bow = Textos a ser analizados
    doc_id = nombre del documento almacenado
    path_glosario = carpeta donde se encuentran los glosarios almacenados

```



```

path_results = carpeta donde se almacenaran los resultados
"""

#inicializacion de las listas a utilizar
glosario = []
glosario_id = []
tokens_glosario = []

#limpieza de los textos
bow = clean_docs(bow)

for filename in os.listdir(path_glosario):
    f2 = open(path_glosario + filename, "r")
    glosario += [f2.read()]
    glosario_id.append(filename)

for g in glosario:
    tokens_glosario += [wordpunct_tokenize(g)]

#creacion del modelo word2vec
w2v_model = models.Word2Vec(sentences = tokens_glosario, size=10, window=3, sg=1,
workers=4, min_count = 1)

query_embedding = [get_embeddings_from_document(w2v_model, g) for g in
tokens_glosario]
vectorized_docs = [get_embeddings_from_document(w2v_model, b) for b in bow]

final = pd.DataFrame()
max_values = pd.DataFrame()

#Para cada elemento en los documentos y para cada elemento del glosario aplicar la
similaridad del coseno
for i, g in enumerate(query_embedding):
    similarities = []
    for j, doc in enumerate(vectorized_docs):
        cs = cosine_similarity(np.array(g).reshape(1,-1), np.array(doc).reshape(1,
-1))
        similarities.append((cs[0][0]))

    #almacenar los resultados
    path_res_glosario = path_results+ "word2vec/word2vec_" + glosario_id[i]
    guardar_resultados(similarities, doc_id, path_res_glosario)
    final[cambiar_singlelabel(glosario_id[i][-13])] = similarities

#Creacion de una estructura tipo dataframe para estructurar los resultados
#La matriz contiene el nombre del documento, la clasificacion real, la probabilidad
de prediccion
#y la clasificacion del modelo.
max_values['Documento'] = doc_id
max_values['Real'] = cambiar_label([i[:-2] for i in doc_id])
max_values['Valor'] = final.max(axis = 1)
max_values['Predicciones'] = final.idxmax(axis=1)

#Se guardan los resultados en clasificacion y en evaluacion
guardar_clasificacion(max_values, doc_id, path_results + "/Clasificacion/word2vec.
txt")
guardar_evaluacion(max_values, path_results + "/Evaluacion/word2vec_eval.txt")

#Llamada al modelo de naive bayes
def naivebayes_model(bow, doc_id, path_glosario, path_results):
    """
    Esta funcion recibe el documento o el BoW, los nombres de los documentos, el path
    donde se encuentran almacenados los glosarios
    el path donde se almacenaran los resultados. Se encarga de generar y aplicar el
    modelo Naive Bayes.

    bow = Textos a ser analizados
    doc_id = nombre del documento almacenado
    path_glosario = carpeta donde se encuentran los glosarios almacenados
    path_results = carpeta donde se almacenaran los resultados
    """

```

```

#Inicializacion de las listas
doc_train = []
label = []
max_values = pd.DataFrame()

for filename in os.listdir(path_glosario):
    f = open(path_glosario+filename,"r")
    files = f.read()
    # Se almacenan todos los documentos en una lista para poder procesarlos
    conjuntamente
    doc_train += [files]
    label += [filename[:-13]]

#Cambiar los nombres de las clases por valores numericos
label = cambiar_label(label)

#Creacion de CountVectorizer para la transformacion de los datos para poder
introducirllos en el modelo
#El modelo recibe como None, puesto que la limpieza del texto se hara con funciones
creadas por nosotras
test = []
cv = CountVectorizer(strip_accents = None, preprocessor = None, stop_words = None)
doc_train = cv.fit_transform(doc_train) #Preparacion de los datos de entrenamiento
bow = clean_docs(bow) #Limpieza de los textos

#Transformar los token en un string, es lo que espera cv.transform como entrada
for s in bow:
    test += ["".join([" "+i if not i.startswith("'") else i for i in s]).strip()]

bow = cv.transform(test) #Preparacion de los datos de test

#Creacion del modelo Naive Bayes e implementacion
naive_bayes = MultinomialNB()
naive_bayes.fit(doc_train, label)
predictions = naive_bayes.predict(bow)
prediction_probabilities = naive_bayes.predict_proba(bow) #Prediccion de las clases
con probabilidades
path_res_glosario = path_results+ "/Naive-Bayes/naivesbayes_results.txt"
guardar_resultados(predictions, doc_id, path_res_glosario)

final = pd.DataFrame(prediction_probabilities)

#Creacion de una estructura tipo dataframe para estructurar los resultados
#La matriz contiene el nombre del documento, la clasificacion real, la probabilidad
de prediccion
#y la clasificacion del modelo.
max_values['Documento'] = doc_id
max_values['Real'] = cambiar_label([i[:-2] for i in doc_id])
max_values['Valor'] = final.max(axis = 1)
max_values['Predicciones'] = predictions

#Se guardan los resultados en clasificacion y en evaluacion
guardar_clasificacion(max_values, doc_id, path_results + "/Clasificacion/NaiveBayes.
txt")
guardar_evaluacion(max_values, path_results + "/Evaluacion/NaiveBayes_eval.txt")

#####
# METODOS PARA PREPROCESAR TEXTOS #
#####

def cambiar_label(label):
    """
    Esta funcion recibe una lista de palabras. Se encarga de transformar los elementos
    de la lista en numeros
    representando una clase.

    label = Lista de palabras
    """

```

```

for j,l in enumerate(label):
    if l.capitalize() == temas[0]:
        label[j] = 0
    elif l.capitalize() == temas[1]:
        label[j] = 1
    elif l.capitalize() == temas[2]:
        label[j] = 2
return label

def cambiar_singlelabel(l):
    """
    Esta funcion recibe una palabras. Se encarga de transformar una palabra en un numero
    representando una clase.

    label = palabras
    """

    if l == temas[0]:
        l = 0
    elif l == temas[1]:
        l = 1
    elif l == temas[2]:
        l = 2
    return l

def clean_docs(docs):
    """
    Esta funcion recibe una lista con documentos. Se encarga de limpiar una lista de
    texto.

    docs = lista con el texto de los documentos
    """

    #Inicializacion de variables
    stemmer = PorterStemmer()
    final = []

    #Para cada documento en la lista de documentos aplicar: tokenizacion, eliminar stop
    words,
    #eliminar si tienen menos de 2 elementos y aplicar radicalizacion
    for doc in docs:
        tokens = wordpunct_tokenize(doc)
        clean = [stemmer.stem(token)
                  for token in tokens
                  if token.lower() not in STOP_WORDS
                  and len(token) > 2
                  and all(c.isalnum() for c in token)]
        final.append([stemmer.stem(word) for word in clean])
    return final

def create_dictionary(path_glosario, pathname= None):
    """
    Esta funcion recibe el path donde se encuentran los glosarios. Se encarga de
    convertir los textos en un diccionario
    y los almacena en corpus

    path_glosario = path donde se encuentra almacenado el glosario
    """

    #se crea un diccionario de corpora
    dictionary = corpora.Dictionary()

    #Para cada palabra del diccionario se separa en tokens
    for filename in os.listdir(path_glosario):
        f2 = open(path_glosario + filename, "r")
        glosario = f2.read()
        tokens = [word for word in glosario.split()]
        dictionary.add_documents([tokens])

```

```

#Guardar el diccionario en memoria
if pathname:
    dictionary.save(pathname+"/corpus.dict")
return dictionary

def create_bow_from_corpus(corpus, dictionary, pathname=None):
    """
    Esta funcion recibe un corpus creado a partir del glosario y un diccionario a partir
    del glosario.
    Se encarga de transformar un diccionario en una Bag of Words (BoW)

    corpus = lista de documentos
    dictionary = diccionario
    """

    #Se crea el Bag of Words mediante la funcion de doc2bow
    bow = [dictionary.doc2bow(text) for text in corpus]

    #Guardar el bow en memoria
    if pathname:
        corpora.MmCorpus.serialize(pathname+'/vsm_docs.mm', bow)
    return bow

#Realizacion del pre-proceso de los textos
def process_text(docs, dictionary):
    """
    Esta funcion recibe una la lista de documentos y el diccionario creado por corpora
    Se encarga de procesar los documentos y crear un bow del corpus a partir de los
    documentos y el diccionario llamando
    a la funcion create_bow_from_corpus.

    docs = lista de documentos
    dictionary = diccionario
    """

    corpus = clean_docs(docs)
    bow = create_bow_from_corpus(corpus, dictionary)
    return bow

#####
# METODOS PARA OBTENER DATOS POR PANTALLA #
#####

# Argparse - Parametros a ser introducidos por el usuario
parser = argparse.ArgumentParser(description="Search by terms")

parser.add_argument('-d',
                    "--directorio",
                    type=str,
                    help="Directorio general donde se encuentran las carpetas con los
                    textos a procesar. El path debe de ser desde la raiz hasta la carpeta Documentos. Ej
                    : ../Document_classification/Documentos")

parser.add_argument('-n',
                    "--nmin",
                    type=int,
                    help="Posicion a partir de la cual se utilizaran los documentos (Por
                    ejemplo, a partir del documento 16)")

parser.add_argument('-r',
                    "--rango",
                    type=int,
                    help="Numero de documentos totales a utilizar para la clasificacion
                    de test (Por ejemplo si deseamos 15 documentos, del 16 al 30)")

parser.add_argument('-m',
                    "--modelo",
                    type=int,
                    help="Modelo a utilizar para el clasificador. 0 = VSM con tf-idf, 1
                    = VSM (word2vec), 2 = Naive Bayes")

```

```

# Parseo de los argumentos
arguments = vars(parser.parse_args())

if arguments['directorio']:
    directorio = arguments['directorio']
else:
    print("ERROR: Porfavor introduzca palabras validas para el directorio")
    exit()
if arguments['nmin']:
    if arguments['nmin'] > 0:
        nmin = arguments['nmin']
    else:
        print("ERROR: El valor de N debe ser mayor que 0")
        exit()
if arguments['rango']:
    if arguments['rango'] > 0:
        modo = arguments['rango']
    else:
        print("ERROR: Introduzca un valor valido mayor que 0")
        exit()
if arguments['modelo']:
    if arguments['modelo'] > 0 and arguments['modelo'] < 3:
        modelo = arguments['modelo']
    else:
        print("ERROR: Introduzca un valor valido mayor que 0 y menor que 2 para un
        modelo valido")
        exit()

clasificador_documentos(directorio, nmin, rango, modelo)

```

Código evaluator.py

```

# Autores: Ana Maria Casado y Ana Sanmartin
#
# Este script calcula la precision, exactitud y sensibilidad/exhaustividad de los
# resultados obtenidos.
# Ademas, los imprime por pantalla, los guarda un documento y crea una matriz de
# confusion que muestra graficamente los resultados.

# Paquetes necesarios para el funcionamiento del programa
from sklearn.metrics import accuracy_score, precision_score, recall_score
import pandas as pd
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

def get_evaluation(results, path_evaluacion, temas):

    print('Accuracy score: ' + str(accuracy_score(results['Real'], results['Predicciones']))) #Calculo de la exactitud
    print('Precision score: ' + str(precision_score(results['Real'], results['Predicciones'], average = 'macro')) #Calculo de la precision
    print('Recall score: ' + str( recall_score(results['Real'], results['Predicciones'], average = 'macro')) #Calculo de la sensibilidad/exhaustividad

    f = open(path_evaluacion, "w")
    f.write('Accuracy score: ' + str(accuracy_score(results['Real'], results['Predicciones']))) + '\n')
    f.write('Precision score: ' + str(precision_score(results['Real'], results['Predicciones'], average = 'macro')) + '\n')
    f.write('Recall score: ' + str( recall_score(results['Real'], results['Predicciones'], average = 'macro')) + '\n')
    f.close()

    cm = confusion_matrix(results['Real'], results['Predicciones']) #Creacion de la matriz de confusion
    sns.heatmap(cm, square=True, annot=True, cmap='Oranges', cbar=False, xticklabels = temas, yticklabels=temas) # Creacion del grafico que muestra los resultados de la matriz de confusion
    plt.ylabel('Clases reales')
    plt.xlabel('Predicciones de clases')
    plt.show()

```