Universidad Politécnica
de Madrid

Escuela Técnica Superior de

Ingenieros Informáticos

Máster Universitario en Inteligencia Artificial

Ingeniería Lingüística

Práctica 2: Clasificador Documental

Alumno 1: Alejandro Francisco Toral

Alumno 2: Enrique Martín López

Madrid, 21 de diciembre de 2020

Contenido

1. Introducción

Esta segunda práctica de Ingeniería Lingüística consiste en la realización de un clasificador documental. A lo largo de esta memoria se encontrarán detallados y justificados todos los pasos que fueron necesarios para la realización del trabajo.

En los apartados 2, 3 y 4 se explican los procedimientos seguidos a la hora de codificar el clasificador de documentos. En el apartado 5 se exponen y evalúan los resultados obtenidos para continuar en el apartado 6 con una serie de conclusiones extraídas a lo largo de la realización de la práctica. Los apartados 7 y 8 contendrán la bibliografía y el cuaderno de Jupyter donde está codificado el trabajo.

1.1. Obtención de documentos

En primer lugar, se han seleccionado 3 temas distintos: política, salud y deportes (los mismos que los propuestos por el profesor de la asignatura). Los temas seleccionados tenían la suficiente disparidad como para ser fácilmente clasificados. De cada uno de los temas se han seleccionado 30 documentos correspondientes a noticias y artículos de opinión y de varios sitios web, periódicos online y agencias de noticias. Todos los documentos corresponden a textos periodísticos por estar al alcance fácilmente a través de Internet. Los documentos son del orden de entre las 400 y las 1000 palabras.

Debido a la pandemia acontecida durante este año, muchas de las noticias y artículos trataban sobre un tema común. Por ende, se ha decidido que muchos de los documentos elegidos sean anteriores al año 2020.

En cuanto al contenido de los documentos, se han buscado aquellos que, aún siendo del mismo tema, aporten variedad al proyecto. Por ejemplo, en el caso de los deportes, se tratan de documentos de todo tipo de estos y no sólo de fútbol o baloncesto.

En un principio, los documentos eran repartidos en 3 carpetas separadas por temas y se seleccionaban, aleatoriamente, la mitad para entrenamiento y la mitad para prueba. Tras una serie de ejecuciones, nos dimos cuenta de que esto podía inducir a error, ya que el glosario de términos variaba mucho debido a la arbitrariedad de los documentos seleccionados: dependiendo de qué 15 se eligieran, el vocabulario iba a ser más o menos denso. Así, los resultados finales eran muy dispares, por lo que no podíamos medir con exactitud cómo de bueno era el clasificador a no ser que cambiásemos la manera de tratar los documentos.

Por este motivo, se decidió variar el modelo inicial: en lugar de seleccionar aleatoriamente los documentos, esta división se haría manualmente, de tal forma que 15 documentos de cada tema estarían en sus correspondientes carpetas y los 15 restantes de cada tópico serían llevados a una carpeta de prueba o test que serviría de repositorio de los documentos que se usarían en la experimentación para probar el clasificador.

Los 15 documentos (45 en total) de la fase de experimentación corresponden a los artículos de mayor densidad para aportar mayor variabilidad al glosario. A continuación, se aportan 3 tablas diferentes con los titulares y los enlaces a cada uno de los artículos seleccionados:

1.1.1. Documentos de deportes

Este subapartado presenta la tabla de documentos numerados relativos al tema de deportes. Están identificados por un cardinal, el titular de la noticia y el enlace donde fue encontrada.

# Documento	Titular	Enlace
1	*Top 10 de propietarios más ricos del mundo de equipos deportivos: meten por error a Amancio Ortega*	https://www.marca.com/buzz/2020/12/14/5fd708d146163fc50c8b4588.html
2	*Sorteo Champions*	https://www.elespanol.com/deportes/futbol/20201214/streaming-directo-sorteo-octavos-final-champions-league/543446000_0.html
3	*El deporte federado vuelve a las canchas*	https://www.diariovasco.com/deportes/mas-deportes/deporte-federado-vuelve-20201214203907-nt.html
4	*Suspenden una prueba ciclista en Navas del Rey porque supuestamente el alcalde convocó a cazadores para impedirla*	https://www.20minutos.es/deportes/noticia/4510184/0/carrera-ciclista-mountain-bike-suspendida-cazadores/
5	*Los deportes de invierno piden al Gobierno que las estaciones de esquí abran en Navidad*	https://as.com/masdeporte/2020/12/02/polideportivo/1606905753_894084.html
6	*Aragón potenciará la presencia de la mujer en todos los ámbitos del deporte*	https://www.heraldo.es/noticias/deportes/2020/12/14/aragon-potenciara-la-presencia-de-la-mujer-en-todos-los-ambitos-del-deporteara-mujer-deporte-1410239.html
7	*Los 'breakers' legitiman su deporte: "Tiene una parte física y otra artística, como muchas disciplinas"*	https://www.marca.com/otros-deportes/2020/12/09/5fcfc673e2704e21958b4617.html
8	*«Si nos venimos a Ibiza es porque aquí nuestro deporte es mejor y más atractivo»*	https://www.diariodeibiza.es/deportes/2020/12/14/venimos-ibiza-deporte-mejor-atractivo/1190012.html
9	*Las "Guerreras" ante su reto más difícil*	https://www.efe.com/efe/espana/deportes/las-guerreras-ante-su-reto-mas-dificil/10006-4409162
10	*0-0. El Leganés resiste el acoso del Celta*	https://www.efe.com/efe/espana/deportes/0-el-leganes-resiste-acoso-del-celta/10006-3843401
11	*Marcelinho Huertas, desde los tres años metiendo canastas*	https://www.efe.com/efe/espana/deportes/marcelinho-huertas-desde-los-tres-anos-metiendo-canastas/10006-4339742
12	*84-71. Serio correctivo del colista Breogán a un Real Madrid para olvidar*	https://www.efe.com/efe/espana/deportes/84-71-serio-correctivo-del-colista-breogan-a-un-real-madrid-para-olvidar/10006-3844380
13	*12-14. Los penaltis decidieron el triunfo de Rusia ante una luchadora España*	https://www.efe.com/efe/espana/deportes/12-14-los-penaltis-decidieron-el-triunfo-de-rusia-ante-una-luchadora-espana/10006-3846645

14	*China honra a Samaranch como una de las figuras clave de su historia reciente*	https://www.efe.com/efe/espana/deportes/china-honra-a-samaranch-como-una-de-las-figuras-clave-su-historia-reciente/10006-3845733
15	*2-0. El Real Madrid exhibe superioridad*	https://www.efe.com/efe/espana/deportes/2-0-el-real-madrid-exhibe-superioridad/10006-4417757
16	*Manuel Martínez ofrece su guía cultural para la sexta semana*	https://www.efe.com/efe/espana/deportes/manuel-martinez-ofrece-su-guia-cultural-para-la-sexta-semana/10006-4225045
17	*Sancionan con 6.000 euros y 2 años sin ir a campos de fútbol a unos padres de jugadores por una pelea en la grada*	https://www.efe.com/efe/espana/deportes/sancionan-con-6-000-euros-y-2-anos-sin-ir-a-campos-de-futbol-unos-padres-jugadores-por-una-pelea/10006-3847809
18	*Marc Gasol asalta el Staples Center; Willy consigue su primer doble-doble*	https://www.efe.com/efe/espana/deportes/marc-gasol-asalta-el-staples-center-willy-consigue-su-primer-doble/10006-3851109
19	*Luis Enrique: "Queremos dar una imagen potente"*	https://www.efe.com/efe/espana/deportes/luis-enrique-queremos-dar-una-imagen-potente/10006-3851998
20	*Bassino y Caviezel ganan; Pinturault comanda solo y Vlhova confirma el liderato*	https://www.efe.com/efe/espana/deportes/bassino-y-caviezel-ganan-pinturault-comanda-solo-vlhova-confirma-el-liderato/10006-4417574
21	*Pau Gasol admite que "sería muy especial" jugar en los Lakers con Marc*	https://www.efe.com/efe/espana/deportes/pau-gasol-admite-que-seria-muy-especial-jugar-en-los-lakers-con-marc/10006-4415491
22	*Real Madrid y Atlético Madrid, Monchengladbach y Atalanta completan octavos*	https://www.efe.com/efe/espana/deportes/real-madrid-y-atletico-monchengladbach-atalanta-completan-octavos/10006-4415261
23	*El Rayo considera una "extorsión orquestada" la denuncia del equipo femenino*	https://www.efe.com/efe/espana/deportes/el-rayo-considera-una-extorsion-orquestada-la-denuncia-del-equipo-femenino/10006-4415005
24	*Los Rockets confirman la llegada de Harden y rechazan hablar de traspaso*	https://www.efe.com/efe/espana/deportes/los-rockets-confirman-la-llegada-de-harden-y-rechazan-hablar-traspaso/10006-4414316
25	*El tsunami de la primera Copa Davis*	https://www.efe.com/efe/espana/deportes/el-tsunami-de-la-primera-copa-davis/10006-4414384
26	*Cristiano deja segundo a Messi y el escándalo llega en París*	https://www.efe.com/efe/espana/deportes/cristiano-deja-segundo-a-messi-y-el-escandalo-llega-en-paris/10006-4414185
27	*Portugal, camino accesible hacia la que podría ser última gran cita Cristiano*	https://www.efe.com/efe/espana/deportes/portugal-camino-accesible-hacia-la-que-podria-ser-ultima-gran-cita-cristiano/10006-4413461

28	*Cuándo, cómo y por qué Diego Maradona se convirtió en un ídolo popular*	https://www.efe.com/efe/espana/deportes/cuando-como-y-por-que-diego-maradona-se-convirtio-en-un-idolo-popular/10006-4413244
29	*El árbitro de la "mano de Dios" vuelve a pitar para homenajear a Maradona*	https://www.efe.com/efe/espana/deportes/el-arbitro-de-la-mano-dios-vuelve-a-pitar-para-homenajear-maradona/10006-4412281
30	*Sordo cede el liderato a Ogier, que queda a un paso del título*	https://www.efe.com/efe/espana/deportes/sordo-cede-el-liderato-a-ogier-que-queda-un-paso-del-titulo/10006-4412280

Tabla 1. Documentos de deportes

1.1.2. Documentos de política

Este subapartado presenta la tabla de documentos numerados relativos al tema de política. Están identificados por un cardinal, el titular de la noticia y el enlace donde fue encontrada.

# Documento	Titular	Enlace
1	*El norte de Europa supera a Madrid en el pulso por el Brexit*	https://elpais.com/ccaa/2020/02/26/madrid/1582742685_730771.html
2	*El supremo decide por unanimidad que se celebre un nuevo juicio a Otegui por el "caso Bateragune"*	https://elpais.com/espana/2020-12-14/el-supremo-decide-por-unanimidad-que-se-celebre-nuevo-juicio-a-otegi-por-el-caso-bateragune-en-audiencia-nacional.html
3	*Sin tiempo para investigar 'Púnica'*	https://elpais.com/ccaa/2020/02/24/madrid/1582564582_253372.html
4	*Díaz Ayuso negocia con Monasterio para bajar impuestos en oposición a Sánchez*	https://elpais.com/ccaa/2020/02/25/madrid/1582624662_580924.html
5	*La UE y el Reino Unido pactan continuar con las negociaciones para evitar un Brexit a las bravas*	https://elpais.com/internacional/2020-12-12/la-ue-y-el-reino-unido-deciden-hoy-por-fin-si-es-posible-evitar-un-brexit-a-las-bravas.html
6	*Las relaciones con Israel dividen a la familia real de Arabia Saudí*	https://elpais.com/internacional/2020-12-13/las-relaciones-con-israel-dividen-a-la-familia-real-de-arabia-saudi.html
7	*Berlín estudia poner bajo vigilancia al partido de derecha radical Alternativa para Alemania*	https://www.abc.es/internacional/abci-berlin-pone-bajo-vigilancia-partido-derecha-radical-alternativa-para-alemania-202012111341_noticia.html
8	*El primer año de Alberto Fernández en Argentina, a imagen y semejanza de Pedro Sánchez*	https://www.abc.es/internacional/abci-primer-alberto-fernandez-argentina-imagen-y-semejanza-pedro-sanchez-202012091427_noticia.html

9	*El exjefe de la Policía apunta en Kitchen a más dirigentes del PP: "Cosidó lo sabía todo"*	https://elpais.com/espana/2020-12-14/el-exjefe-de-la-policia-apunta-en-kitchen-a-mas-dirigentes-del-pp-cosido-lo-sabia-todo.html
10	*Juanma Moreno ganaría en Andalucía por 3,1 puntos sobre el PSOE y volvería a sumar con Cs y Vox*	https://www.elespanol.com/espana/andalucia/20201214/juanma-moreno-andalucia-psoe-volveria-cs-vox/543446111_0.html
11	*La renovación judicial abre una nueva brecha en la coalición del Gobierno*	https://www.elespanol.com/espana/politica/20201209/sanchez-justifica-envio-inmigrantes-peninsula-empatizar-canarias/542197306_0.html
12	*Sánchez justifica el envío opaco de inmigrantes a la Península: "Hay que empatizar con Canarias"*	https://www.lavanguardia.com/politica/20201214/6118224/renovacion-judicial-abre-nueva-brecha-coalicion-gobierno.html
13	*El PSC asegura que "votar a ERC es votar a Puigdemont"*	https://www.lavanguardia.com/politica/20201214/6119470/psc-asegura-votar-erc-votar-puigdemont-elecciones-cataluna.html
14	*La renovación judicial abre una nueva brecha en la coalición del Gobierno*	https://www.lavanguardia.com/politica/20201214/6118224/renovacion-judicial-abre-nueva-brecha-coalicion-gobierno.html
15	*Los intentos de los servicios secretos de Reino Unido de interferir en la política de América Latina en los años 60*	https://www.bbc.com/mundo/noticias-america-latina-55274993
16	*Londres da un paso y elimina las cláusulas que violaban el acuerdo del Brexit*	https://www.elconfidencial.com/mundo/europa/2020-12-08/brexit-boris-johnson-acuerdo-complicado-ue_2863532/
17	*Hacia una visión estratégica europea común*	https://www.elconfidencial.com/mundo/2020-11-30/union-europea-vision-estrategica-comune_2851559/
18	*Las masivas protestas en Francia contra la ley de seguridad se saldan con 50 detenidos*	https://www.elconfidencial.com/mundo/2020-11-28/francia-nuevos-disturbios-protestas-ley-seguridad_2852267/
19	*Un tribunal de apelaciones desestima la última demanda de Trump en Pensilvania*	https://www.elconfidencial.com/mundo/2020-11-27/tribunal-apelaciones-desestima-ultima-demanda-trump-en-pensilvania_2851979/
20	*El Consejo de Europa pide a España reforzar la lucha contra la violencia machista*	https://www.elconfidencial.com/mundo/europa/2020-11-25/consejo-europa-pide-espana-reforzar-lucha-violencia-machista_2847007/

21	*La UE celebra el quinto aniversario del Acuerdo de París por el cambio climático*	https://www.elconfidencial.com/mundo/europa/2020-12-12/ue-celebra-quinto-aniversario-acuerdo-paris_2869836/
22	*EEUU ejecuta la novena pena de muerte de este año el Día de los Derechos Humanos*	https://www.elconfidencial.com/mundo/2020-12-11/estados-unidos-ejecuta-novena-pena-muerte-brandon-bernard-derechos-humanos_2867728/
23	*La Comisión refuerza la preparación ante un 'no acuerdo' con Londres el 1 de enero*	https://www.elconfidencial.com/mundo/europa/2020-12-10/la-comision-refuerza-la-preparacion-ante-un-no-acuerdo-con-londres-el-1-de-enero_2866343/
24	*Aparecen 2.600 votos sin contabilizar en Georgia durante el recuento de las elecciones*	https://www.elconfidencial.com/mundo/2020-11-17/elecciones-estados-unidos-recuento-georgia-votos_2836243/
25	*Los letrados del Congreso rechazan la comisión de investigación sobre el Rey émerito*	https://www.larazon.es/espana/20201214/crlwze4dujamlmecarcqccbnjq.html
26	*La UE y Reino Unido se dan una oportunidad para sortear el divorcio caótico el 31 de diciembre*	https://www.larazon.es/internacional/20201213/wzz5o2ke6fbjnhbejmhhudakrq.html
27	*Los partidos presentan hoy sus diferencias ante el informe sobre crisis*	https://www.efe.com/efe/espana/politica/los-partidos-presentan-hoy-sus-diferencias-ante-el-informe-sobre-crisis/10002-3844589
28	*Laya asegura que Iglesias firmó a título individual la Declaración de La Paz*	https://www.larazon.es/espana/20201214/zgjlqdnyn5f7rjflvrhiryo3ui.htmlj
29	*El CGPJ desafía al Gobierno con más nombramientos*	https://www.larazon.es/espana/20201214/3cjyj7x4k5gnpan4vpk2mz2fd4.html
30	*Felipe VI, un Rey solo bajo el pulso Sánchez-Casado*	https://www.larazon.es/espana/20201213/tz554pdmfvaonn3zsjwamiiiqa.html

Tabla 2. Documentos de política

1.1.3. Documentos de salud

Este subapartado presenta la tabla de documentos numerados relativos al tema de salud. Están identificados por un cardinal, el titular de la noticia y el enlace donde fue encontrada.

# Documento	Titular	Enlace
1	*Esclerosis Múltiple: se cumplen cincuenta años de su abordaje en España*	https://www.efesalud.com/esclerosis-multiple-cincuenta-anos-abordaje-espana/
2	*Sida, un día mundial bajo la sombra maldita de la COVID*	https://www.efesalud.com/sida-dia-mundial-covid-19/
3	*¿Cómo han afectado la pandemia y el confinamiento a los pacientes neurológicos?*	https://www.efesalud.com/pandemia-confinamiento-pacientes-neurologicos/
4	*Depresión grave y ansiedad, efectos psicológicos de la pandemia*	https://www.efesalud.com/depresion-grave-ansiedad-efectos-psicologia-pandemia/
5	*Informe ASISTO: un nuevo modelo asistencial para el largo superviviente oncológico*	https://www.efesalud.com/largos-supervivientes-oncologicos-informe-asisto-nuevo-modelo-asistencial/
6	*Salud mental en el cáncer: el papel de la psicooncología en la asistencia integral*	https://www.efesalud.com/salud-mental-cancer-papel-psicooncologia/
7	*Diagnóstico precoz frente al cáncer de pulmón*	https://www.efesalud.com/diagnostico-precoz-dia-cancer-pulmon/
8	*Asma 360: abordaje integral para su control*	https://www.efesalud.com/educacion-diagnostico-claves-mejorar-calidad-vida-asmaticos/
9	*¿Tengo la gripe o la COVID?*	https://www.efesalud.com/tengo-gripe-o-covid/
10	*La investigación es vida y la esperanza de los pacientes con cáncer*	https://www.efesalud.com/investigacion-cancer-pacientes/
11	*El 90 % de los españoles mayores de 60 años desconoce qué son las Valvulopatías*	https://www.efesalud.com/cardiologia-semana-valvulopatias/
12	*La mortalidad por infarto de miocardio se duplica durante la pandemia*	https://www.efesalud.com/infarto-miocardio-mortalidad-duplica-pandemia/
13	*Frenillo sublingual corto, una patología fácil de solucionar en los bebés*	https://www.efesalud.com/frenillo-sublingual-anquiloglosia-bebes/

14	*Un nuevo mecanismo de rejuvenecimiento celular podría revertir la artrosis*	https://www.efesalud.com/artrosis-rejuvenecimiento-celular-investigacion
15	*Ojo con las fracturas, se teme un rebrote*	https://www.efesalud.com/fracturas-rebrote-mayores/
16	*El miedo a engordar tras el confinamiento puede disparar aún más los Trastornos de la Conducta Alimentaria*	https://www.efesalud.com/trastornos-conducta-alimentaria-dia-internacional-confinamiento/
17	*"Meningitis, cerrando el círculo", un documental para prevenir y concienciar*	https://www.efesalud.com/meningitis-cerrando-circulo-documental-prevenir-concienciar/
18	*Informe enfermedades neurológicas: los ingresos hospitalarios, en alza*	https://www.efesalud.com/informe-enfermedades-neurologicas-hospitales-sen
19	*Nuevos tratamientos contra la migraña*	https://www.efesalud.com/nuevos-tratamientos-contra-la-migrana/
20	*Se puede erradicar la Hepatitis C*	https://www.efesalud.com/erradicar-hepatitis+C
21	*Migraña: La voz de los pacientes ante una enfermedad que cambia la vida*	https://www.efesalud.com/migrana-convivir-pacientes-enfermedad
22	*Voluntarios contra la malaria*	https://www.efesalud.com/voluntarios-contra-la-malaria/
23	*Las nuevas infecciones por sida bajaron un 5,3 % en 2017 y las muertes un 5 por ciento*	https://www.efesalud.com/nuevas-infecciones-muertes-sida-bajaron-2017/
24	*Nuevo brote de ébola en el Congo, dos casos y 17 muertes sospechosas*	https://www.efesalud.com/nuevo-brote-ebola-congo/
25	*La falta de financiación y formación frenan la lucha contra el cáncer en Latinoamérica*	https://www.efesalud.com/falta-financiacion-formacion-frenan-lucha-cancer-latinoamerica/
26	*Vivir con "huesos de cristal" no frena los deseos por sobresalir*	https://www.efesalud.com/huesos-de-cristal-mexicano-enfermedad
27	*Párkinson: Controlan los efectos motores con ultrasonido HIFU y sin cirugía*	https://www.efesalud.com/parkinson-control-efectos-motores-ultrasonido
28	*Ocho de cada 10 personas con problemas de salud mental no tienen empleo*	https://www.efesalud.com/salud-mental-empleo

29	*Trastornos de la conducta alimentaria, silencioso sufrimiento por autoexigencia*	https://www.efesalud.com/trastornos-conducta-alimentaria-silencioso-sufrimiento/
30	*Protectores solares: El riesgo de conservar el del año pasado*	https://www.efesalud.com/protectores-solares-riesgos/

Tabla 3. Documentos de salud

2. Constructor del glosario

En este apartado se explica la obtención del glosario de términos necesario para clasificar los documentos.

2.1. Primer modelo: *stemming*

En una primera instancia se pensó que la mejor manera de implementar el glosario era mediante la extracción de las raíces de las palabras más repetidas a lo largo del texto (después de haber procesado rodos los documentos). Para esto se usó la técnica *stemming*.

Debido a los comentarios del profesor realizados en clase y un posterior correo enviado, se decidió eliminar el glosario por raíces por el siguiente motivo: el *steeming* es válido cuando la variabilidad lingüística es alta porque la cantidad de documentos de la colección sea excesivamente grande (del orden de cientos de miles e incluso millones de documentos). En colecciones más pequeñas como en el caso de esta práctica, el *steeming* ayuda poco ya que se corre el riesgo de que todas las variaciones lingüísticas de una palabra tengan una baja frecuencia.

2.2. Modelo final: sustracción terminológica

Para la creación del glosario con el que se entrenará el modelo, hemos optado con una creación semiautomática en la cual se genera un glosario de términos a partir de los documentos, con una serie de parámetros que ajustamos de forma manual y con los cuales somos capaces de obtener distintos resultados.

El procedimiento que se sigue para la creación del glosario es el siguiente:

1. Se cargan los documentos de entrenamiento por temáticas y se inicializan los parámetros que utilizamos para crear el glosario. El funcionamiento de estos parámetros se explica en el paso 6.
2. Para cada una de las temáticas se procesan los distintos documentos y se detectan las distintas palabras que hay en dichos documentos con los siguientes criterios:
    a. No se incluyen en dicha lista palabras vacías o *stopwords* tanto en español como en inglés, además de palabras que nosotros hemos considerado que no aportan información como por ejemplo *Sun* (Palabra que se refería a *The Sun*, una revista inglesa). Tampoco se incluyen símbolos y signos de puntuación.
    b. Las palabras se recogen siempre con y sin tildes, y siempre y cuando tengan más de 3 caracteres.
3. Una vez recogido el listado de palabras para cada una de las temáticas, se contabiliza cuantas veces aparece cada término en todos los documentos de dicha temática.
4. A partir de un parámetro que establecemos nosotros para cada una de las temáticas, excluimos aquellos términos que no alcancen un mínimo de veces que aparezca en los documentos.
5. Juntamos los términos de todas las temáticas. Si un término que se va a incluir de una temática ya está incluido (por ejemplo, porque aparezca en otra temática), no se incluye (porque ya está incluido).

Ahora, se va a explicar cómo se ha codificado cada uno de los pasos:

Paso 1: inicializamos los parámetros que se utilizarán en el paso 6:

```
#Parámetros usados
min_deportes = 9
min_politica = 10
min_salud = 9
```

Fragmento de código 1. Parámetros del glosario

Después hemos creado la siguiente función, con la que se cargan los archivos que se especifican en la variable filenames (que son las rutas de los archivos que se desean cargar), con codificación utf-8 para poder leer las tildes, y se eliminan los saltos de línea y los retornos de carro que pudiese haber en el texto. La función devuelve una lista con cada uno de los documentos que se han cargado.

```
#Función de carga de los documentos
def load_documents(filenames):
    return [io.open(name, 'r', encoding='utf-8').read().replace("\r", "").replace("\n", "") for name in filenames]
```

Fragmento de código 2. Función de carga de documentos

De esa forma podemos cargar los documentos de cada una de las temáticas de la siguiente forma, haciendo uso de la librería glob, que detecta las rutas de los archivos que hay en un determinado directorio y crea una lista con dichas rutas:

```
deportes = load_documents(sorted(glob.glob('Deportes/*')))
```

De esta forma, en la variable deportes, habrá una lista con cada documento de dicha temática en cada posición de la lista.

Hacemos lo mismo para el resto de los temas:

```
politica = load_documents(sorted(glob.glob('Política/*')))
```

```
salud = load_documents(sorted(glob.glob('Salud/*')))
```

Paso 2:

Haciendo uso de la siguiente función para crear el listado de palabras o tokens para cada temática:

```
#Funcion tokenize (extracción de términos relevantes)
def tokenize(text):

    #Lista de palabras vacías (stopwords): combinación entre archivo .json + lista de palabras extra + stopwords en español e inglés del paquete stopwords
    add_stopwords = ['tan', 'través', 'aún', 'sun', 'ser', 'estar', 'tener', 'haber', 'efe', 'además', 'aun']
    stopwords_json = json.load(open('stop_words_spanish.json',"rb"), encoding="utf-8")
    stopwords_spanish = stopwords.words('spanish') + add_stopwords + stopwords.words('english') + stopwords_json

    #Lista de signos de puntuación
    non_words = list(punctuation)
    non_words.extend(['¿', '¡','"', '“','«','º','ºc','»','‘','’','…','-','–','—','ª'])
    non_words.extend(map(str,range(10)))

    #Obtención de los términos de los textos sin los signos de puntuación y demás elementos que no son palabras
    text = ''.join([c.lower() for c in text if c not in non_words])

    #Eliminación de las stopwords de la lista anterior y devolución de la lista final de términos
    tokens_with_stopwords = re.findall("[A-Za-zÀ-ÖØ-öø-ÿ]{3,}", text)
    tokens = [token for token in tokens_with_stopwords if token not in stopwords_spanish]
    return tokens
```

Fragmento de código 3. Función Tokenize

En la cual se cargan las distintas palabras vacías que se desean excluir, por un lado, haciendo uso de la variable ‘add_stopwords’ que son las variables que nosotros añadimos de forma manual, stopwords_json, que corresponden a palabras vacías presentes en un archivo json, que se adjunta con

la entrega, y se juntan todas esas palabras junto a las stopwords en español e inglés proporcionadas por la librería nltk.

También se excluyen los símbolos de puntuación y los números del 0 al 9. Por último se buscan las distintas palabras con o sin tilde y con 3 o más caracteres a partir de la expresión regular que se especifica en la creación de la variable tokens_with_stopwords, sobre la cual se eliminarán el listado de stopwords creado anteriormente en la variable stopwords_spanish y se devolverá dicho listado.

Paso 3:

Se hace uso de la función tokenize explicada en el paso anterior para crear un listado de las distintas palabras presentes en los distintos documentos y se contabiliza las veces que aparecen dichas palabras en los documentos. Esto se especifica en la siguiente función:

```
#Funcion usada para contar el número de apariciones de las palabras en un texto
def count_words(documents):

    #Lista de palabras encontradas en los documentos tras pasar por el tokenizer
    wordstring = ''
    for document in documents:
        wordstring += " ".join(str(item) for item in tokenize(document))

    wordlist = wordstring.split()

    #Lista de frecuencias de las palabras
    wordfreq = []
    for w in wordlist:
        wordfreq.append(wordlist.count(w))

    return wordlist, wordfreq
```

Fragmento de código 4. Función de conteo de palabras

Paso 4:

Se hace uso de la función especificada anteriormente para cada una de las temáticas y se excluyen aquellos términos que no aparezcan al menos el número de veces que se especifica en los parámetros del paso 1. Una vez excluídos dichos términos, tendríamos creado el glosario de cada una de las temáticas.

```
#Se seleccionan los documentos de test para hacer el glosario sólo con ellos
wordlist, wordfreq = count_words(deportes)

#Se forma la lista de términos
deportes_new_vocab = []
for word, count in zip(wordlist,wordfreq):

    #Se comprueba que los términos aparezcan un mínimo de veces y no estén ya incluídos previamente en el glosario
    if(count >= min_deportes and word not in deportes_new_vocab):

        #La lista final será un glosario de cada tema sin palabras repetidas
        deportes_new_vocab.append(word)
```

Fragmento de código 5. Obtención del glosario del tema "deportes"

```
#Se seleccionan los documentos de test para hacer el glosario sólo con ellos
wordlist, wordfreq = count_words(politica)

#Se forma la lista de términos
politica_new_vocab = []
for word, count in zip(wordlist,wordfreq):

    #Se comprueba que los términos aparezcan un mínimo de veces y no estén ya incluídos previamente en el glosario
    if(count >= min_politica and word not in politica_new_vocab):

        #La lista final será un glosario de cada tema sin palabras repetidas
        politica_new_vocab.append(word)
```

Fragmento de código 6. Obtención del glosario del tema "política"

```
#Se seleccionan los documentos de test para hacer el glosario sólo con ellos
wordlist, wordfreq = count_words(salud)

#Se forma la lista de términos
salud_new_vocab = []
for word, count in zip(wordlist,wordfreq):

    #Se comprueba que los términos aparezcan un mínimo de veces y no estén ya incluídos previamente en el glosario
    if(count >= min_salud and word not in salud_new_vocab):

        #La lista final será un glosario de cada tema sin palabras repetidas
        salud_new_vocab.append(word)
```

Fragmento de código 7. Obtención del glosario del tema "salud"

Paso 5:

Por último, se añaden los términos de cada temática a una variable 'vocabulario' que será el glosario que se utilizará. En esta variable si un término ya está añadido, no se vuelve a añadir.

```
#La lista "vocabulario" será el resultado de sumar las tres listas anteriores sin palabras repetidas
vocabulario = deportes_new_vocab
vocabulario = vocabulario + [token for token in politica_new_vocab if token not in vocabulario]
vocabulario = vocabulario + [token for token in salud_new_vocab if token not in vocabulario]
vocabulario.sort()

print(vocabulario)
```

Fragmento de código 8. Obtención del glosario final

De modo que, por ejemplo, teniendo los parámetros inicializados de la siguiente forma:

```
#Parámetros usados
min_deportes = 9
min_politica = 10
min_salud = 9
```

Obtenemos los siguientes glosarios, que serán los utilizados en los siguientes pasos de la construcción del clasificador.

2.3. Glosario de deportes

Letra	Palabra	Frecuencia de aparición
A	Actividad	9
	Anna	9
	Atlético	11
	Años	21
B	Baloncesto	14
C	China	13
	Competición	12

	Conjunto	9
	Cuarto	9
D	Deporte	29
	Deportivo	11
	Diego	9
E	Encuentro	9
	Equipo	23
	Equipos	12
	España	24
	Español	20
F	Final	20
	Frente	9
G	Grupo	11
H	Historia	9
	Horas	9
J	Juan	10
	Jugadores	11
L	Liga	9
M	Madrid	28
	Maradona	24
	Minutos	11
P	Partido	10
	Popular	10
	Presidente	10
	Prueba	9
R	Real	19
S	Rival	11
	Samaranch	14

Tabla 4. Glosario de deportes

Adicionalmente se incluyen más términos, correspondientes a la variabilidad lingüística demandada en el enunciado. La lista de términos añadidos a mano es la siguiente:

['actividades', 'competiciones', 'conjuntos', 'cuartos', 'deportes', 'deportiva', 'deportivas', 'deportivos', 'encuentros', 'española', 'españolas', 'españoles', 'finales', 'frentes', 'grupos', 'historias', 'hora', 'jugador', 'jugadora', 'jugadoras', 'ligas', 'minuto', 'partidos', 'presidentes', 'pruebas', 'reales', 'rivales']

2.4. Glosario de política

Letra	**Palabra**	**Frecuencia de aparición**
A	Acuerdo	46
	Audiencia	14
	Año	10
	Años	21
B	Brexit	10
	Británico	12
	Bruselas	10
C	Carlos	11
	Casado	18
	Caso	22
	CGPJ	19

	Comisión	11
	Congreso	16
	Consejo	11
	Crisis	10
D	Días	13
E	Empresas	17
	Enero	11
	España	13
	Europea	15
	Europeo	10
F	Fernández	11
	Formación	11
	Funciones	10
G	General	11
	Gobierno	72
I	Interior	15
	Israel	11
J	Johnson	12
	Judicial	16
	Juez	11
L	Ley	15
	Londres	14
	Lunes	10
M	Madrid	27
	Mercado	11
	Mes	11
	Ministro	23
	Moncloa	10
N	Nacional	16
O	Oposición	14
P	Pablo	13
	Pacto	17
	Partido	18
	País	10
	Policía	11
	Política	18
	Presidente	37
	Problema	11
	PSOE	21
R	Reino	25
	Renovar	11
	Rey	16
S	Saudí	12
	Semana	10
	Supremo	12
	Sánchez	36
T	Tribunal	14
U	Unidas	19
	Unido	23

Tabla 5. Glosario de política

Adicionalmente se incluyen más términos, correspondientes a la variabilidad lingüística demandada en el enunciado. La lista de términos añadidos a mano es la siguiente:

[‘acuerdos', ‘audiencias’, ‘británicos’, ‘británicas’, ‘británica’, ‘casos’, ‘comisiones’, ‘congresos’, ‘consejos’, ‘día’, ‘empresa’, ‘europeas’, ‘europeos’, ‘formaciones’, ‘función’, ‘generales’, ‘gobiernos’, ‘judiciales’, ‘jueces’, ‘jueza’, ‘juezas’, ‘leyes’, ‘mercados’, ‘meses’, ‘ministros’, ‘ministra’, ‘ministras’, ‘nacionales’, ‘oposiciones’, ‘pactos’, ‘partidos’, ‘países’, ‘policías’, ‘problemas’, ‘reina’, ‘reyes’, ‘reinas’, ‘semana’, ‘saudíes’, ‘tribunales’]

2.5. Glosario de salud

Letra	Palabra	Frecuencia de aparición
A	Asociación	9
	Atención	12
	Año	18
	Años	28
C	Cáncer	36
	Carlos	11
	Caso	15
	Casos	27
	Covid	22
D	Diagnostico	16
	Día	22
	Días	13
	Dolor	12
E	Ecuador	9
	Enfermedad	45
	Enfermedades	12
	España	9
	Española	12
	Evitar	10
	Expertos	12
	Explica	17
F	Falta	13
	Forma	13
	Frenillo	12
G	Gripe	18
H	Hospital	12
I	Indica	9
	Infecciones	12
L	Lucha	9
M	Malaria	9
	Mental	23
	Meses	15
	Migraña	10
	Millones	9
	Mujeres	14
	Mundial	11
N	Niños	9
O	Objetivo	11

P	Paciente	16
	Pacientes	32
	Pandemia	13
	Patología	10
	País	10
	Países	12
	Personas	37
	Piel	12
	Prevención	12
	Problema	12
	Problemas	22
	Profesionales	11
	Protección	14
R	Recursos	13
S	Salud	44
	Sanitarios	12
	Sida	11
	Sociedad	13
	Solar	16
	Síntomas	11
T	Tipo	13
	Trastornos	13
	Tratamiento	17
	Tratamientos	11
U	Universidad	9
V	Vida	19
	VIH	20
	Virus	12

Tabla 6. Glosario de salud

Adicionalmente se incluyen más términos, correspondientes a la variabilidad lingüística demandada en el enunciado. La lista de términos añadidos a mano es la siguiente:

[‘asociaciones’, ‘casos’, ‘diagnósticos’, ‘dolores’, ‘experto’, ‘experta’, ‘expertas’, ‘hospitales’, ‘infección’, ‘luchas’, ‘mentales’, ‘mes’, ‘mujer’, ‘mundiales’, ‘niño’, ‘objetivos’, ‘pandemias’, ‘patologías’, ‘persona’, ‘profesional’, ‘recurso’, ‘sanitario’, ‘síntoma’, ‘tipos’, ‘trastorno’, ‘universidades’]

2.6. Glosario de términos

Finalmente, el glosario que se utiliza en el clasificador es el resultante de la combinación de los tres glosarios anteriores (deportes, salud y política), junto con las variaciones lingüísticas añadidas posteriormente tras una primera revisión de los glosarios temáticos. El conjunto de palabras final es el siguiente:

['actividad', 'actividades', 'acuerdo', 'acuerdos', 'anna', 'asociaciones', 'asociación', 'atención', 'atlético', 'audiencia', 'audiencias', 'año', 'años', 'baloncesto', 'brexit', 'británica', 'británicas', 'británico', 'británicos', 'bruselas', 'carlos', 'casado', 'caso', 'casos', 'cgpj', 'china', 'comisiones', 'comisión', 'competiciones', 'competición', 'congreso', 'congresos', 'conjunto', 'conjuntos', 'consejo', 'consejos', 'covid', 'crisis', 'cuarto', 'cuartos', 'cáncer', 'deporte', 'deportes', 'deportiva', 'deportivas', 'deportivo', 'deportivos', 'diagnóstico', 'diagnósticos', 'diego', 'dolor', 'dolores', 'día', 'días', 'ecuador',

'empresa', 'empresas', 'encuentro', 'encuentros', 'enero', 'enfermedad', 'enfermedades', 'equipo', 'equipos', 'españa', 'español', 'española', 'españolas', 'españoles', 'europea', 'europeas', 'europeo', 'europeos', 'evitar', 'experta', 'expertas', 'experto', 'expertos', 'explica', 'falta', 'fernández', 'final', 'finales', 'forma', 'formaciones', 'formación', 'frenillo', 'frente', 'frentes', 'funciones', 'función', 'general', 'generales', 'gobierno', 'gobiernos', 'gripe', 'grupo', 'grupos', 'historia', 'historias', 'hora', 'horas', 'hospital', 'hospitales', 'indica', 'infecciones', 'infección', 'interior', 'israel', 'johnson', 'juan', 'judicial', 'judiciales', 'jueces', 'juez', 'jueza', 'juezas', 'jugador', 'jugadora', 'jugadoras', 'jugadores', 'ley', 'leyes', 'liga', 'ligas', 'londres', 'lucha', 'luchas', 'lunes', 'madrid', 'malaria', 'maradona', 'mental', 'mentales', 'mercado', 'mercados', 'mes', 'meses', 'migraña', 'millones', 'ministra', 'ministras', 'ministro', 'ministros', 'minuto', 'minutos', 'moncloa', 'mujer', 'mujeres', 'mundial', 'mundiales', 'nacional', 'nacionales', 'niño', 'niños', 'objetivo', 'objetivos', 'oposiciones', 'oposición', 'pablo', 'paciente', 'pacientes', 'pacto', 'pactos', 'pandemia', 'pandemias', 'partido', 'partidos', 'patología', 'patologías', 'país', 'países', 'persona', 'personas', 'piel', 'policía', 'policías', 'política', 'popular', 'presidente', 'presidentes', 'prevención', 'problema', 'problemas', 'profesional', 'profesionales', 'protección', 'prueba', 'pruebas', 'psoe', 'real', 'reales', 'recurso', 'recursos', 'reina', 'reinas', 'reino', 'renovar', 'rey', 'reyes', 'rival', 'rivales', 'salud', 'samaranch', 'sanitario', 'sanitarios', 'saudí', 'saudíes', 'semana', 'sida', 'sociedad', 'solar', 'supremo', 'sánchez', 'síntoma', 'síntomas', 'tipo', 'tipos', 'trastorno', 'trastornos', 'tratamiento', 'tratamientos', 'tribunal', 'tribunales', 'unidas', 'unido', 'universidad', 'universidades', 'vida', 'vih', 'virus']

Hay términos que no son sustantivos comunes que hemos decidido dejar por ser lo suficientemente discriminantes en ciertos temas. Por ejemplo, se puede observar que una palabra muy repetida en el tema deportivo es “Maradona”, en referencia al futbolista Diego Armando Maradona; o en política, las palabras “Casado” o “Sánchez”, que hacen referencia a los apellidos de dirigentes políticos españoles. También decidimos dejar palabras abreviadas y acrónimos por el mismo motivo. Por ejemplo: “CPGJ” (Consejo General del Poder Judicial) o “VIH” (virus de la inmunodeficiencia humana).

Para la variabilidad lingüística se han incluido las formas en femenino, masculino y/o plural de todas aquellas palabras a las que les faltara.

3. TF-IDF

En un clasificador es fundamental conocer cómo de importantes son las palabras dentro de cada documento en una colección. Para ello hacemos uso de la medida vista en clase: Tf-idf (*Term frequency – Inverse document frequency*, en castellano, frecuencia de término – frecuencia inversa de documento). Esta medida se utiliza a menudo como un factor de ponderación en la recuperación de información y la minería de texto. El valor tf-idf aumenta proporcionalmente al número de veces que una palabra aparece en el documento, pero es compensada por la frecuencia de la palabra en la colección de documentos, lo que permite manejar el hecho de que algunas palabras son generalmente más comunes que otras [5].

3.1. Funcionamiento

En esta práctica se quiere clasificar una serie de documentos según su tema principal: deportes, política y salud.

Supongamos que en los documentos de deportes se repiten de manera habitual las palabras “el”, “fútbol” y “lesión”, mientras que en los de salud, esas palabras que más aparecen son “el”, “virus” y “mal”. Para saber a qué tema pertenece cada documento bastaría con ver cuáles de esas palabras coinciden en cada uno. Sin embargo, para diferenciarlos aún más, debemos contar el número de veces que cada término ocurre en cada documento y sumarlos: el número de veces que un término ocurre en un documento se denomina su frecuencia de término (**tf**).

Sin embargo, como el término "el" es tan común, esto provocará que se destaquen incorrectamente documentos que utilizan de casualidad la palabra "el" con más frecuencia, sin conceder suficiente peso a los demás términos significativos. El término "el" no es una buena palabra clave para distinguir documentos relevantes y no relevantes, a diferencia del resto de palabras. Por lo tanto, se incorpora un factor de frecuencia inversa de documento (**idf**) que atenúa el peso de los términos que ocurren con mucha frecuencia en la colección de documentos e incrementa el peso de los términos que ocurren pocas veces.

Matemáticamente hablando, tf-idf funciona de la siguiente forma:

- **Tf(t, d)**: es sencillamente la frecuencia bruta del término t en el documento d, o sea, el número de veces que el término t ocurre en el documento d.
- **Idf(t, D):** la frecuencia inversa de documento es una medida de si el término es común o no, en la colección de documentos. Se obtiene dividiendo el número total de documentos por el número de documentos que contienen el término, y se toma el logaritmo de ese cociente:

  $$idf(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

  Donde |D| es el número de documentos de la colección y el cociente anterior es el número de documentos donde aparece el término t.

Finalmente, el resultado de tf-idf se calcula como:

$$tfidf(t, d, D) = tf(i, d) \times idf(t, D)$$

Para implementar esta medida en nuestra práctica hemos utilizado la librería de Sklearn: *sklearn.feature_selection.text*. A continuación, se muestra el fragmento de código donde se ha implementado dicha función:

```
vectorizer = TfidfVectorizer(
                analyzer = 'word',
                tokenizer = tokenize,
                vocabulary = vocabulario)

X_Train_TF = vectorizer.fit_transform(X_Train)
X_Test_TF =  vectorizer.transform(X_Test)
```

Fragmento de código 9. Implementación de la métrica tf-idf

Únicamente son necesarias las funciones:

- **TFidfVectorizer**. Convierte una colección de documentos sin procesar en una matriz de funciones TF-IDF [6]. Utiliza los siguientes parámetros:
    - **Analyzer:** indica qué tipo de elemento va a analizar. En este caso, palabras.
    - **Tokenizer:** determina que los términos sean independientes unos de otros
    - **Vocabulary:** utiliza el glosario determinado anteriormente.
- **Fit_transform.** Aprende el vocabulario e idf, y devuelve una matriz de documentos y términos. Se le pasa el conjunto de documentos entrenamiento como argumento.
- **Transform.** Transforma los documentos en una matriz de documentos y términos. Se le pasa como argumento el conjunto de los documentos de prueba que el clasificador no conoce.

Después, el conjunto de los documentos procesados a través de tf-idf se ponen a disposición del clasificador, explicado en el siguiente apartado.

4. Clasificación

El modelo que hemos utilizado para poder clasificar los documentos por temas es el KNeighborsClassifier, haciendo uso de la implementación proporcionada por la librería sci-kit learn [1].

Una de las razones principales de la elección de KNN como clasificador principal fue el conocimiento previo del mismo, ya que lo hemos utilizado en diversas ocasiones en otras asignaturas.

Este clasificador consiste en, para un dato nuevo, asignarle la clase que tengan la mayoría de sus k vecinos más próximos, donde k es un parámetro escogido por el usuario. En este caso, un documento se clasificará de un tema, el cual sea el mayoritario de los k documentos más similares al mismo.

Además, este clasificador proporciona una probabilidad para cada una de las clases, algo que deseamos ya que es un requisito de esta práctica. Esta probabilidad consiste en que, si tenemos K vecinos más próximos y no todos ellos son del mismo tema, entonces se asigna una probabilidad a cada uno de los temas en función del número de vecinos dentro de esos K vecinos más próximos que pertenezcan a cada uno de los distintos temas.

Como medida de similitud, idealmente queremos utilizar la medida de similitud coseno definida por la siguiente expresión, donde A y B serían los vectores que definen los documentos que se van a comparar y θ es en ángulo que separa a dichos vectores (cuanto mayor sea este ángulo, más distintos son ambos vectores):

$$\cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

Como el modelo seleccionado, no contempla dicha medida como medida de distancia para comparar con los vecinos, pero sí la distancia euclídea, lo que hemos hecho ha sido la siguiente equivalencia, partiendo de que normalizando los datos (haciendo que la suma de sus valores sume 1), obtenemos lo mismo utilizando la distancia euclídea que con la del coseno:

Partiendo de la medida del coseno y si tenemos que:

$$(\sum x_i^2 = \sum y_i^2 = 1)$$

Entonces:

$$\cos(x, y) = \frac{\sum x_i y_i}{\sqrt{\sum x_i^2 \sum y_i^2}} = \sum x_i y_i$$

Con la distancia euclídea podemos hacer la siguiente conversión:

$$
\begin{aligned}
||x-y||^2 &= \sum (x_i - y_i)^2 \\
&= \sum (x_i^2 + y_i^2 - 2x_i y_i) \\
&= \sum x_i^2 + \sum y_i^2 - 2\sum x_i y_i \\
&= 1 + 1 - 2\cos(x,y) \\
&= 2(1 - \cos(x,y))
\end{aligned}
$$

En donde se puede apreciar que hay una relación linear directa entre la distancia euclídea y la similitud coseno, siempre y cuando X e Y estén normalizados (En nuestro caso los datos de entrada) [2] [3] [4]. Por lo que lo primero que hacemos antes de pasar a la implementación del modelo es normalizar nuestros datos:

```
X_Train_TF = preprocessing.normalize(X_Train_TF)
X_Test_TF = preprocessing.normalize(X_Test_TF)
```

En cuanto a la implementación, en primer lugar, le pasamos a este modelo el conjunto de datos de entrenamiento, formado por un par de variables, X_Train_TF (matriz de valores de tdf-idf para los documentos de entrenamiento, donde cada fila son los valores de tdf-idf de dicho documento), e Y_Train (clases reales de cada uno de los documentos). Esto lo hacemos a través de la función fit proporcionada por el clasificador.

```
knn = KNeighborsClassifier(n_neighbors=n,weights='uniform', metric='euclidean').fit(X_Train_TF, Y_Train)
```

Una vez realizado esto, utilizamos la función score del clasificador para comprobar cómo de bien ha asignado las etiquetas a cada uno de los documentos en la fase de entrenamiento. Esta función devuelve el porcentaje de acierto en un valor de 0 a 1.

```
score_train = knn.score(X_Train_TF,Y_Train)
```

Después, haciendo uso de las variables X_Test_TF, Y_Test, análogas a las variables usadas en la fase de entrenamiento, pero esta vez con la información de los documentos de test, llamamos a la función predict_proba que devuelve para cada documento, las probabilidades de pertenecer a cada una de las clases, y una vez más a la función score para valorar cómo de bien clasifica cada uno de los documentos en sus respectivos temas.

```
proba_test = knn.predict_proba(X_Test_TF)
score_test = knn.score(X_Test_TF,Y_Test)
```

Por último, mostramos los valores de obtenidos en las funciones anteriores como se muestra a continuación:

```
temas = {0:'Deportes', 1:'Politica', 2:'Salud'}


t = PrettyTable(['Documento', 'Probabilidad (%)', 'Tema'])
for document_name, proba in zip(filenames_test,proba_test):
    t.add_row([document_name.replace("Test\\", ""), max(proba)*100, temas[np.argmax(proba)]])

print(t)
print('Train Accuracy:', score_train * 100, '%')
print('Test Accuracy:', score_test * 100, '%')
```

Fragmento de código 10. Resultados

Los distintos resultados obtenidos cambiando los parámetros en la creación del glosario y cambiando el número de vecinos del modelo se discutirán en el siguiente apartado.

5. Experimentación

Una vez finalizada la codificación e implementación del clasificador, nos dispusimos a probar los parámetros importantes.

5.1. Obtención del mejor número mínimo de apariciones de un término en cada tema

Comenzamos la serie de experimentos definiendo el parámetro de los vecinos como $n = 1$, y probamos distintos valores para conseguir el mejor glosario posible. El objetivo de esta parte de la experimentación era conseguir un glosario restrictivo en cada una de las partes. Además, los glosarios obtenidos para cada uno de los temas debían ser de una longitud parecida para evitar tener más términos de un tema que de otro y hacer que el clasificador tienda al fallo.

En una primera criba, se descubrió que el vocabulario de los documentos relacionados con la política era más numeroso que aquellos relacionados con el deporte o la salud. Por este motivo, el número mínimo de apariciones de las palabras para considerarlas importantes en los documentos de política debía ser mayor que en los otros dos temas.

También se determinó que el número mínimo de apariciones en cada uno de los temas debía ser ≥ 3, pues si era inferior, los glosarios contenían palabras muy redundantes o generales.

En una segunda criba, los números mínimos fueron variando en un intervalo de $(4 \leq n \leq 10)$, siendo *n* el número mínimo apariciones de cada término. Tras una serie de pruebas, comprobamos que los mejores valores para cada uno de los mínimos eran:

- Nº mínimo de apariciones en los documentos de deportes (*min_deportes*) = 9
- Nº mínimo de apariciones en los documentos de política (*min_política*) = 10
- Nº mínimo de apariciones en los documentos de salud (*min_salud*) = 9

Con valores más pequeños se obtenían resultados realmente parecidos a aquellos obtenidos con los anteriores números, pero la cantidad de palabras en el glosario era muy superior. Decidimos buscar un óptimo algo más alejado de los valores pequeños para evitar tener un glosario excesivamente grande. El glosario resultante se puede consultar en la sección 2.6 de esta memoria.

5.2. Obtención del mejor valor de vecindad

Tras obtener el mejor glosario, se procedió directamente a probar distintos valores de *n*, siendo *n* el número de vecinos del algoritmo K-Nearest-Neighbor.

Al tratarse de un clasificador relativamente sencillo (tanto por la cantidad de documentos a clasificar como por la cantidad de entrenamiento), se acotó la búsqueda a un rango de valores relativamente pequeño, por lo que no tardamos en dar con el óptimo. Estos valores fueron en un rango de $(2 \leq n \leq 10)$.

Tras una serie de pruebas, el mejor resultado lo obtuvimos con $n = 4$. En el apartado siguiente se muestra la tabla de coincidencias y el porcentaje de exactitud en entrenamiento y en pruebas.

Una vez fijado el número de vecinos, se probó a variar de nuevo las ocurrencias mínimas anteriormente descritas, pero los resultados no mejoraban. Por ende, se decidió fijar los parámetros anteriores y proceder a la discusión de los mejores resultados obtenidos.

6. Discusión de los resultados

Con los siguientes valores:

- Nº mínimo de apariciones en los documentos de deportes (min_deportes) = 9
- Nº mínimo de apariciones en los documentos de política (min_política) = 10
- Nº mínimo de apariciones en los documentos de salud (min_salud) = 9
- Nº de vecinos óptimo (n) = 4

Se han obtenido los siguientes resultados en el clasificador de documentos:

Documento	**Probabilidad (%)**	**Tema**
4-test-politica.txt	100	Política
10-test-salud.txt	100	Salud
13-test-salud.txt	100	Salud
12-test-politica.txt	100	Política
3-test-politica.txt	75	Política
14-test-salud.txt	100	Salud
7-test-deportes.txt	75	Deportes
9-test-politica.txt	100	Política
14-test-deportes.txt	100	Deportes
5-test-deportes.txt	100	Deportes
8-test-salud.txt	75	Salud
2-test-deportes.txt	75	Deportes
6-test-deportes.txt	100	Deportes
13-test-politica.txt	75	Política
8-test-politica.txt	100	Política
11-test-salud.txt	100	Salud
9-test-salud.txt	100	Salud
8-test-deportes.txt	100	Deportes
2-test-salud.txt	100	Salud
3-test-deportes.txt	100	Deportes
2-test-politica.txt	100	Política
10-test-politica.txt	50	Deportes
6-test-salud.txt	100	Salud
5-test-politica.txt	100	Política
4-test-salud.txt	100	Salud
6-test-politica.txt	100	Política
11-test-politica.txt	50	Política
7-test-politica.txt	100	Política
10-test-deportes.txt	100	Deportes
15-test-politica.txt	100	Política
15-test-deportes.txt	75	Deportes
1-test-deportes.txt	100	Deportes
14-test-politica.txt	100	Política
9-test-deportes.txt	100	Deportes
3-test-salud.txt	100	Salud
12-test-salud.txt	75	Salud
13-test-deportes.txt	100	Deportes
1-test-salud.txt	100	Salud

7-test-salud.txt	100	Salud
15-test-salud.txt	100	Salud
12-test-deportes.txt	100	Deportes
1-test-politica.txt	100	Política
11-test-deportes.txt	75	Deportes
5-test-salud.txt	100	Salud
4-test-deportes.txt	75	Deportes

Tabla 7. Resultados por documento

Con un porcentaje de aciertos en entrenamiento del 100% y un porcentaje de aciertos durante las pruebas de un 97,77%.

El único documento que da fallo utilizando todos estos parámetros es "*10-test-politica.txt*". El resto de los documentos son clasificados perfectamente, la mayoría con una coincidencia del 100%.

Aunque la mayoría de artículos elegidos eran muy distintos entre sí, existían algunos que podían ser confundidos fácilmente por el clasificador, como, por ejemplo, aquellos artículos de deportes que hablan de partes médicos de los jugadores o artículos de salud que hablan de la financiación de un hospital por parte de un ayuntamiento. A pesar de ello, el clasificador ha conseguido clasificar satisfactoriamente la gran mayoría de documentos según el tema al que pertenecen.

En líneas generales estamos contentos con el trabajo realizado, puesto que conseguir tal porcentaje de acierto en nuestro primer clasificador no parecía tarea sencilla.

Suponemos que tal porcentaje no solo se debe a la bondad de nuestro clasificador, sino también a la calidad de los documentos elegidos para este trabajo: los temas eran fácilmente clasificables (tenían las suficientes diferencias como para discernir un tema de otro). Además, dentro de los documentos escogidos también había diversidad, lo que seguramente haya facilitado aún más la clasificación por parte del programa.

7. Conclusiones

En este apartado se presentan una serie de conclusiones relativas al trabajo y opiniones de los alumnos.

Este trabajo ha supuesto un cambio con relación a otras asignaturas del máster: es un proyecto que, además de suponer investigación, sirve como ejemplo de proyecto real que nos podemos encontrar en el mundo laboral. Enfrentarnos a esta práctica ha sido a la par desafiante y entretenido. Teníamos un objetivo, que era hacer un clasificador, y una serie de pautas concretas: las herramientas las ponemos nosotros. Esta variedad y libertad de elección en los trabajos es necesaria para dar dinamismo a la asignatura.

De hecho, la cantidad de opciones a elegir a la hora de implementar la práctica nos hace ser críticos con nosotros mismos y hace que nos demos cuenta de las limitaciones, así como de nuestros puntos fuertes a la hora de enfrentar este tipo de trabajo.

Es importante destacar que el tiempo de implementación ha sido superior al esperado debido a algunos contratiempos encontrados:

- Mala elección del clasificador
- Cambios en la formalización del glosario
- Cambios en la manera de seleccionar los documentos

A pesar de ello, se ha conseguido entregar en la fecha límite propuesta por el profesor, con margen suficiente como para redactar la memoria justificando cada paso dado.

7.1. Futuras líneas de trabajo

Aunque el resultado del clasificador haya sido muy bueno, en una investigación más detallada se podrían variar los artículos utilizados como conjunto de prueba, para ver si es verdad que se comporta tan bien en todos los ámbitos. Se recuerda que los artículos elegidos eran muy diferentes entre sí y por ello el resultado obtenido. Quizás con unos documentos algo más parecidos, pero con temáticas distintas, sean necesarios algunos cambios para llegar a obtener el mismo resultado o, al menos, uno parecido.

Un punto importante a la hora de realizar este tipo de trabajos es la comparación de técnicas. Debido a la falta de tiempo, a la complejidad que supone y a que no era el objetivo de la práctica, no se ha implementado ningún clasificador extra utilizando nuevas técnicas además del entregado, pero consideramos que es algo interesante y que puede llegar a ser un buen proyecto de investigación.

Además de comparar distintas mecánicas de clasificación, otra propuesta de implementación futura es el aumento de los documentos del clasificador. A pesar de que 30 documentos de cada tema hayan sido suficientes para implementar un clasificador exitoso, pensamos que aumentar mucho más el número de documentos puede resultar interesante. Así podríamos usar nuevas técnicas y comparar con las que ya tenemos (es el caso del desechado stemming para conseguir el glosario).

8. Bibliografía

8.1. Fuentes de los artículos

www.marca.com

www.elespanol.com

www.diariovasco.com

www.20minutos.es

www.as.com

www.heraldo.es

www.diariodeibiza.es

www.efe.com

www.elpais.com

www.abc.es

www.lavanguardia.com

www.bbc.com

www.elconfidencial.com

www.larazon.es

www.efesalud.com

8.2. Referencias

[1] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

[2] https://stackoverflow.com/questions/34144632/using-cosine-distance-with-scikit-learn-kneighborsclassifier

[3] https://stats.stackexchange.com/questions/299013/cosine-distance-as-similarity-measure-in-kmeans

[4] https://stackoverflow.com/questions/46409846/using-k-means-with-cosine-similarity-python

[5] https://es.wikipedia.org/wiki/Tf-idf

[6] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

9. Cuaderno

En este apartado se encuentra adjunto el cuaderno de Jupyter donde se ha implementado el clasificador. Contiene el código comentado, así como los mejores resultados obtenidos.

El cuaderno de Jupyter se ha ejecutado utilizando Jupyter Notebook, en local. Para poder ejecutarse correctamente, es necesario que el cuaderno esté incluido en una misma carpeta con:

- Una carpeta "Deportes" que contenga los documentos de entrenamiento del tema "deportes". El nombre de cada elemento de esta carpeta será "*n-entrenamiento-deportes.txt*", siendo n un número natural utilizado para ordenar los documentos. Todos los documentos serán del tipo .txt.
- Una carpeta "Política" que contenga los documentos de entrenamiento del tema "política". El nombre de cada elemento de esta carpeta será "n-entrenamiento-politica.txt"
- Una carpeta "Salud" que contenga los documentos de entrenamiento del tema "salud". El nombre de cada elemento de esta carpeta será "n-entrenamiento-salud.txt"
- Una carpeta "Test" que contenga un conjunto de documentos de los tres temas anteriores. El nombre de cada elemento de esta carpeta será "n-test-tema".
- Un fichero .json que incluye las palabras vacías (*stopwords*) en español.

INGENIERÍA LINGÜÍSTICA. PRÁCTICA 2

En este cuaderno se encuentra toda la implementación de la práctica 2 de la asignatura de Ingeniería Lingüística del máster de Inteligencia Artifical de la UPM.

El cuaderno se va a dividir en las siguientes partes:

- 1. Imports
- 2. Funciones de carga y procesado de documentos
- 3. Implementación del algoritmo de clasificación
- 4. Pruebas y evaluación

1. Imports

Se han utilizado diversas librerías de python:

In [1]:

```
import glob
import io
import json
import numpy as np
import math
import re
import random
from sklearn import preprocessing  # to normalise existing X
from collections import Counter
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neighbors import DistanceMetric
from prettytable import PrettyTable
```

In [2]:

```
import nltk
from nltk.corpus import stopwords
from nltk import word_tokenize
from nltk.data import load
from string import punctuation

from sklearn.feature_extraction.text import TfidfVectorizer
nltk.download('stopwords')
nltk.download('punkt')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\alexb\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\alexb\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[2]:

```
True
```

2. Funciones de carga y procesado de documentos

En este apartado se detallan las funciones utilizadas para el preprocesado de los textos y la extracción de vocabulario.

2.1. Parámetros

Estos parámetros se utilizan en distintas funciones del código. Se ajustan para probar la mejora o empeoramiento de nuestro modelo. Estos parámetros son:

- **Número mínimo de aparición**: son tres parámetros distintos que se usan para determinar cuál es el número mínimo de apariciones que debe tener una palabra en un conjunto de textos de un mismo tema para ser considerada importante o de interés para la clasificación
  - **min_deportes**: número mínimo de veces que debe aparecer una palabra en los textos de deportes para que sea considerada
  - **min_politica**: número mínimo de veces que debe aparecer una palabra en los textos de política para que sea considerada
  - **min_salud**: número mínimo de veces que debe aparecer una palabra en los textos de salud para que sea considerada

In [3]:

```
#Parámetros usados
min_deportes = 9
min_politica = 10
min_salud = 9
```

2.2. Funciones de procesado de documentos y obtención del glosario

A continuación, se encuentran las funciones que se han utilizado para obtener el glosario de términos.

In [4]:

```
#Función de carga de los documentos
def load_documents(filenames):
    return [io.open(name, 'r', encoding='utf-8').read().replace("\r", "").replace("\n", "")
```

In [5]:

```
#Función que comprueba la etiqueta original de cada documento
#Usada para comprobar la bondad de nuestro clasificador y expresar el porcentaje de acierto
def check_original_labels(filenames):
    labels = []

    #Se aprovecha que cada documento tiene en su nombre la etiqueta a la que pertenece orig
    for name in filenames:
        if 'deportes' in name:
            labels.append(0)
        elif 'politica' in name:
            labels.append(1)
        elif 'salud' in name:
            labels.append(2)
        else:
            labels.append(random.randint(0,3))

    #Devolución de la lista de etiquetas reales
    return labels
```

In [6]:

```
#Funcion tokenize (extracción de términos relevantes)
def tokenize(text):

    #Lista de palabras vacías (stopwords): combinación entre archivo .json + lista de palab
    add_stopwords = ['tan', 'través', 'aún', 'sun', 'ser', 'estar', 'tener', 'haber', 'efe'
    stopwords_json = json.load(open('stop_words_spanish.json',"rb"), encoding="utf-8")
    stopwords_spanish = stopwords.words('spanish') + add_stopwords + stopwords.words('engli

    #Lista de signos de puntuación
    non_words = list(punctuation)
    non_words.extend(['¿', '¡','”', '“','«','º','ºc','»','‘','’','…','–','—','—','ª'])
    non_words.extend(map(str,range(10)))

    #Obtención de los términos de los textos sin los signos de puntuación y demás elementos
    text = ''.join([c.lower() for c in text if c not in non_words])

    #Eliminación de las stopwords de la lista anterior y devolución de la lista final de té
    tokens_with_stopwords = re.findall("[A-Za-zÀ-ÖØ-öø-ÿ]{3,}", text)
    tokens = [token for token in tokens_with_stopwords if token not in stopwords_spanish]
    return tokens
```

In [7]:

```
#Funcion usada para contar el número de apariciones de las palabras en un texto
def count_words(documents):

    #Lista de palabras encontradas en los documentos tras pasar por el tokenizer
    wordstring = ''
    for document in documents:
        wordstring += " ".join(str(item) for item in tokenize(document))

    wordlist = wordstring.split()

    #Lista de frecuencias de las palabras
    wordfreq = []
    for w in wordlist:
        wordfreq.append(wordlist.count(w))

    return wordlist, wordfreq
```

2.3. Glosario de deportes

In [8]:

```
#Carga de los documentos pertenecientes al apartado de deportes
deportes = load_documents(sorted(glob.glob('Deportes/*')))
```

In [9]:

```
#Se seleccionan los documentos de test para hacer el glosario sólo con ellos
wordlist, wordfreq = count_words(deportes)

#Se forma la lista de términos
deportes_new_vocab = []
for word, count in zip(wordlist,wordfreq):

    #Se comprueba que los términos aparezcan un mínimo de veces y no estén ya incluídos pre
    if(count >= min_deportes and word not in deportes_new_vocab):

        #La lista final será un glosario de cada tema sin palabras repetidas
        deportes_new_vocab.append(word)

variab_deportes = ['actividades', 'competiciones', 'conjuntos', 'cuartos', 'deportes', 'dep
                   'deportivos', 'encuentros', 'española', 'españolas', 'españoles', 'final
                   'historias', 'hora', 'jugador', 'jugadora', 'jugadoras', 'ligas', 'minut
                   'pruebas', 'reales', 'rivales']
deportes_new_vocab = deportes_new_vocab + variab_deportes

deportes_new_vocab.sort()
print(deportes_new_vocab)
```

```
['actividad', 'actividades', 'anna', 'atlético', 'años', 'baloncesto', 'chin
a', 'competiciones', 'competición', 'conjunto', 'conjuntos', 'cuarto', 'cuar
tos', 'deporte', 'deportes', 'deportiva', 'deportivas', 'deportivo', 'deport
ivos', 'diego', 'encuentro', 'encuentros', 'equipo', 'equipos', 'españa', 'e
spañol', 'española', 'españolas', 'españoles', 'final', 'finales', 'frente',
'frentes', 'grupo', 'grupos', 'historia', 'historias', 'hora', 'horas', 'jua
n', 'jugador', 'jugadora', 'jugadoras', 'jugadores', 'liga', 'ligas', 'madri
d', 'maradona', 'minuto', 'minutos', 'partido', 'partidos', 'popular', 'pres
idente', 'presidentes', 'prueba', 'pruebas', 'real', 'reales', 'rival', 'riv
ales', 'samaranch']
```

2.4. Glosario de política

In [10]:

```
#Carga de los documentos pertenecientes al apartado de política
politica = load_documents(sorted(glob.glob('Política/*')))
```

In [11]:

```
#Se seleccionan los documentos de test para hacer el glosario sólo con ellos
wordlist, wordfreq = count_words(politica)

#Se forma la lista de términos
politica_new_vocab = []

for word, count in zip(wordlist,wordfreq):

    #Se comprueba que los términos aparezcan un mínimo de veces y no estén ya incluídos pre
    if(count >= min_politica and word not in politica_new_vocab):

        #La lista final será un glosario de cada tema sin palabras repetidas
        politica_new_vocab.append(word)


variab_politica = ['acuerdos', 'audiencias', 'británicos', 'británicas', 'británica', 'caso
                   'consejos', 'día', 'empresa', 'europeas', 'europeos', 'formaciones', 'fu
                   'gobiernos', 'judiciales', 'jueces', 'jueza', 'juezas', 'leyes', 'mercad
                   'ministra', 'ministras', 'nacionales', 'oposiciones', 'pactos', 'partido
                   'problemas', 'reina', 'reyes', 'reinas', 'saudíes', 'tribunales']
politica_new_vocab = politica_new_vocab + variab_politica

politica_new_vocab.sort()
print(politica_new_vocab)
```

```
['acuerdo', 'acuerdos', 'audiencia', 'audiencias', 'año', 'años', 'brexit',
'británica', 'británicas', 'británico', 'británicos', 'bruselas', 'carlos',
'casado', 'caso', 'casos', 'cgpj', 'comisiones', 'comisión', 'congreso', 'co
ngresos', 'consejo', 'consejos', 'crisis', 'día', 'días', 'empresa', 'empres
as', 'enero', 'españa', 'europea', 'europeas', 'europeo', 'europeos', 'ferná
ndez', 'formaciones', 'formación', 'funciones', 'función', 'general', 'gener
ales', 'gobierno', 'gobiernos', 'interior', 'israel', 'johnson', 'judicial',
'judiciales', 'jueces', 'juez', 'jueza', 'juezas', 'ley', 'leyes', 'londre
s', 'lunes', 'madrid', 'mercado', 'mercados', 'mes', 'meses', 'ministra', 'm
inistras', 'ministro', 'ministros', 'moncloa', 'nacional', 'nacionales', 'op
osiciones', 'oposición', 'pablo', 'pacto', 'pactos', 'partido', 'partidos',
'país', 'países', 'policía', 'policías', 'política', 'presidente', 'problem
a', 'problemas', 'psoe', 'reina', 'reinas', 'reino', 'renovar', 'rey', 'reye
s', 'saudí', 'saudíes', 'semana', 'supremo', 'sánchez', 'tribunal', 'tribuna
les', 'unidas', 'unido']
```

2.5. Glosario de salud

In [12]:

```
#Carga de los documentos pertenecientes al apartado de salud
salud = load_documents(sorted(glob.glob('Salud/*')))
```

In [13]:

```
#Se seleccionan los documentos de test para hacer el glosario sólo con ellos
wordlist, wordfreq = count_words(salud)

#Se forma la lista de términos
salud_new_vocab = []
for word, count in zip(wordlist,wordfreq):

    #Se comprueba que los términos aparezcan un mínimo de veces y no estén ya incluídos pre
    if(count >= min_salud and word not in salud_new_vocab):

        #La lista final será un glosario de cada tema sin palabras repetidas
        salud_new_vocab.append(word)

variab_salud = ['asociaciones', 'casos', 'diagnósticos', 'dolores', 'experto', 'experta', '
                'luchas', 'mentales', 'mes', 'mujer', 'mundiales', 'niño', 'objetivos', 'pa
                'persona', 'profesional', 'recurso', 'sanitario', 'síntoma', 'tipos', 'tras
salud_new_vocab = salud_new_vocab + variab_salud

salud_new_vocab.sort()
print(salud_new_vocab)
```

```
['asociaciones', 'asociación', 'atención', 'año', 'años', 'carlos', 'caso',
'casos', 'casos', 'covid', 'cáncer', 'diagnóstico', 'diagnósticos', 'dolor',
'dolores', 'día', 'días', 'ecuador', 'enfermedad', 'enfermedades', 'españa',
'española', 'evitar', 'experta', 'expertas', 'experto', 'expertos', 'explic
a', 'falta', 'forma', 'frenillo', 'gripe', 'hospital', 'hospitales', 'indic
a', 'infecciones', 'infección', 'lucha', 'luchas', 'malaria', 'mental', 'men
tales', 'mes', 'meses', 'migraña', 'millones', 'mujer', 'mujeres', 'mundia
l', 'mundiales', 'niño', 'niños', 'objetivo', 'objetivos', 'paciente', 'paci
entes', 'pandemia', 'pandemias', 'patología', 'patologías', 'país', 'paíse
s', 'persona', 'personas', 'piel', 'prevención', 'problema', 'problemas', 'p
rofesional', 'profesionales', 'protección', 'recurso', 'recursos', 'salud',
'sanitario', 'sanitarios', 'sida', 'sociedad', 'solar', 'síntoma', 'síntoma
s', 'tipo', 'tipos', 'trastorno', 'trastornos', 'tratamiento', 'tratamiento
s', 'universidad', 'universidades', 'vida', 'vih', 'virus']
```

2.6. Glosario final

In [14]:

```
#La lista "vocabulario" será el resultado de sumar las tres listas anteriores sin palabras
vocabulario = deportes_new_vocab
vocabulario = vocabulario + [token for token in politica_new_vocab if token not in vocabula
vocabulario = vocabulario + [token for token in salud_new_vocab if token not in vocabulario
vocabulario.sort()

print(vocabulario)
```

```
['actividad', 'actividades', 'acuerdo', 'acuerdos', 'anna', 'asociaciones',
'asociación', 'atención', 'atlético', 'audiencia', 'audiencias', 'año', 'año
s', 'baloncesto', 'brexit', 'británica', 'británicas', 'británico', 'britán
cos', 'bruselas', 'carlos', 'casado', 'caso', 'casos', 'cgpj', 'china', 'com
isiones', 'comisión', 'competiciones', 'competición', 'congreso', 'congreso
s', 'conjunto', 'conjuntos', 'consejo', 'consejos', 'covid', 'crisis', 'cuar
to', 'cuartos', 'cáncer', 'deporte', 'deportes', 'deportiva', 'deportivas',
'deportivo', 'deportivos', 'diagnóstico', 'diagnósticos', 'diego', 'dolor',
'dolores', 'día', 'días', 'ecuador', 'empresa', 'empresas', 'encuentro', 'en
cuentros', 'enero', 'enfermedad', 'enfermedades', 'equipo', 'equipos', 'espa
ña', 'español', 'española', 'españolas', 'españoles', 'europea', 'europeas',
'europeo', 'europeos', 'evitar', 'experta', 'expertas', 'experto', 'experto
s', 'explica', 'falta', 'fernández', 'final', 'finales', 'forma', 'formacion
es', 'formación', 'frenillo', 'frente', 'frentes', 'funciones', 'función',
'general', 'generales', 'gobierno', 'gobiernos', 'gripe', 'grupo', 'grupos',
'historia', 'historias', 'hora', 'horas', 'hospital', 'hospitales', 'indic
a', 'infecciones', 'infección', 'interior', 'israel', 'johnson', 'juan', 'ju
dicial', 'judiciales', 'jueces', 'juez', 'jueza', 'juezas', 'jugador', 'juga
dora', 'jugadoras', 'jugadores', 'ley', 'leyes', 'liga', 'ligas', 'londres',
'lucha', 'luchas', 'lunes', 'madrid', 'malaria', 'maradona', 'mental', 'ment
ales', 'mercado', 'mercados', 'mes', 'meses', 'migraña', 'millones', 'minist
ra', 'ministras', 'ministro', 'ministros', 'minuto', 'minutos', 'moncloa',
'mujer', 'mujeres', 'mundial', 'mundiales', 'nacional', 'nacionales', 'niñ
o', 'niños', 'objetivo', 'objetivos', 'oposiciones', 'oposición', 'pablo',
'paciente', 'pacientes', 'pacto', 'pactos', 'pandemia', 'pandemias', 'partid
o', 'partidos', 'patología', 'patologías', 'país', 'países', 'persona', 'per
sonas', 'piel', 'policía', 'policías', 'política', 'popular', 'presidente',
'presidentes', 'prevención', 'problema', 'problemas', 'profesional', 'profes
ionales', 'protección', 'prueba', 'pruebas', 'psoe', 'real', 'reales', 'recu
rso', 'recursos', 'reina', 'reinas', 'reino', 'renovar', 'rey', 'reyes', 'ri
val', 'rivales', 'salud', 'samaranch', 'sanitario', 'sanitarios', 'saudí',
'saudíes', 'semana', 'sida', 'sociedad', 'solar', 'supremo', 'sánchez', 'sín
toma', 'síntomas', 'tipo', 'tipos', 'trastorno', 'trastornos', 'tratamient
o', 'tratamientos', 'tribunal', 'tribunales', 'unidas', 'unido', 'universida
d', 'universidades', 'vida', 'vih', 'virus']
```

3. Implementación del algoritmo de clasificación

In [15]:

```
#Conjunto de documentos de entrenamiento del clasificador
X_Train = deportes + politica + salud
Y_Train = [0] * len(deportes) + [1] * len(politica) + [2] * len(salud)

#Conjunto de documentos de test (prueba) del clasificador
filenames_test = glob.glob('Test/*')
X_Test = load_documents(filenames_test)
Y_Test = check_original_labels(filenames_test)

#Se aleatorizan todos los documentos, tanto los de entrenamiento como los osmetidos a las p
c = list(zip(X_Train, Y_Train))

random.shuffle(c)

X_Train, Y_Train = zip(*c)

d = list(zip(X_Test, Y_Test, filenames_test))

random.shuffle(d)

X_Test, Y_Test, filenames_test = zip(*d)
```

3.1. TF_IDF

In [16]:

```
#Se aplica la métrica tf-idf vista en clase
vectorizer = TfidfVectorizer(
                analyzer = 'word',
                tokenizer = tokenize,
                vocabulary = vocabulario)

X_Train_TF = vectorizer.fit_transform(X_Train)
X_Test_TF =  vectorizer.transform(X_Test)
```

3.2. KNN y resultados

In [17]:

```
n = 4
X_Train_TF = preprocessing.normalize(X_Train_TF)
X_Test_TF = preprocessing.normalize(X_Test_TF)

knn = KNeighborsClassifier(n_neighbors=n,weights='uniform', metric='euclidean').fit(X_Train

score_train = knn.score(X_Train_TF,Y_Train)
proba_test = knn.predict_proba(X_Test_TF)
score_test = knn.score(X_Test_TF,Y_Test)


temas = {0:'Deportes', 1:'Politica', 2:'Salud'}


t = PrettyTable(['Documento', 'Probabilidad (%)', 'Tema'])
for document_name, proba in zip(filenames_test,proba_test):
    t.add_row([document_name.replace("Test\\", ""), max(proba)*100, temas[np.argmax(proba)]

print(t)
print('Train Accuracy:', score_train * 100, '%')
print('Test Accuracy:', score_test * 100, '%')
```

```
+----------------------+------------------+----------+
|      Documento       | Probabilidad (%) |   Tema   |
+----------------------+------------------+----------+
| 4-test-politica.txt  |      100.0       | Politica |
|  10-test-salud.txt   |      100.0       |  Salud   |
|  13-test-salud.txt   |      100.0       |  Salud   |
| 12-test-politica.txt |      100.0       | Politica |
| 3-test-politica.txt  |       75.0       | Politica |
|  14-test-salud.txt   |      100.0       |  Salud   |
| 7-test-deportes.txt  |       75.0       | Deportes |
| 9-test-politica.txt  |      100.0       | Politica |
| 14-test-deportes.txt |      100.0       | Deportes |
| 5-test-deportes.txt  |      100.0       | Deportes |
|   8-test-salud.txt   |       75.0       |  Salud   |
| 2-test-deportes.txt  |       75.0       | Deportes |
| 6-test-deportes.txt  |      100.0       | Deportes |
| 13-test-politica.txt |       75.0       | Politica |
| 8-test-politica.txt  |      100.0       | Politica |
|  11-test-salud.txt   |      100.0       |  Salud   |
|   9-test-salud.txt   |      100.0       |  Salud   |
| 8-test-deportes.txt  |      100.0       | Deportes |
|   2-test-salud.txt   |      100.0       |  Salud   |
| 3-test-deportes.txt  |      100.0       | Deportes |
| 2-test-politica.txt  |      100.0       | Politica |
| 10-test-politica.txt |       50.0       | Deportes |
|   6-test-salud.txt   |      100.0       |  Salud   |
| 5-test-politica.txt  |      100.0       | Politica |
|   4-test-salud.txt   |      100.0       |  Salud   |
| 6-test-politica.txt  |      100.0       | Politica |
| 11-test-politica.txt |       50.0       | Politica |
| 7-test-politica.txt  |      100.0       | Politica |
| 10-test-deportes.txt |      100.0       | Deportes |
| 15-test-politica.txt |      100.0       | Politica |
| 15-test-deportes.txt |       75.0       | Deportes |
| 1-test-deportes.txt  |      100.0       | Deportes |
| 14-test-politica.txt |      100.0       | Politica |
```

```
| 9-test-deportes.txt  |      100.0       | Deportes |
|   3-test-salud.txt   |      100.0       |  Salud   |
|  12-test-salud.txt   |       75.0       |  Salud   |
| 13-test-deportes.txt |      100.0       | Deportes |
|   1-test-salud.txt   |      100.0       |  Salud   |
|   7-test-salud.txt   |      100.0       |  Salud   |
|  15-test-salud.txt   |      100.0       |  Salud   |
| 12-test-deportes.txt |      100.0       | Deportes |
| 1-test-politica.txt  |      100.0       | Politica |
| 11-test-deportes.txt |       75.0       | Deportes |
|   5-test-salud.txt   |      100.0       |  Salud   |
| 4-test-deportes.txt  |       75.0       | Deportes |
+----------------------+------------------+----------+
Train Accuracy: 100.0 %
Test Accuracy: 97.77777777777777 %
```