


Práctica 2: Clasificador Documental

Ingeniería Lingüística

Luna Jiménez Fernández
Alejandro Muñoz Navarro

MUIA - UPM
Curso: 2020/2021

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

El objetivo de esta práctica es el desarrollo de un **clasificador de textos** capaz de clasificar cualquier documento en uno de tres temas: **deportes**, **política** o **salud**.

Para esto se utilizará un conjunto de **glosarios** extraídos previamente de una fuente de documentos y se probarán distintos pares de métricas y modelos, con el fin de identificar el mejor clasificador de entre todas las combinaciones probadas.

La estructura de la memoria es la siguiente:

- En el **Capítulo 2** se expondrá la metodología seguida para la **selección y almacenamiento de los artículos** y se mostrará la lista de artículos seleccionados (con enlaces a cada uno de los artículos y a las fuentes de las que han sido extraídos).
- En el **Capítulo 3** se expondrá la metodología seguida para la **extracción de los glosarios de cada tema**, centrándose además en la implementación realizada y el uso del extractor, y exponiendo los glosarios obtenidos.
- En el **Capítulo 4** se hablará sobre el clasificador documental, explicando algunos conceptos necesarios (las **métricas y modelos** que serán utilizados) y pasando a explicar la **metodología** seguida por el clasificador. Además, se comentará la **implementación y uso** de susodicho clasificador.
- En el **Capítulo 5** se expondrán los **resultados** obtenidos por el clasificador documental, planteando una serie de **hipótesis**, realizando un **análisis** de los mismos y extrayendo una serie de **conclusiones**.
- En el **Capítulo 6** se citaran las **conclusiones** alcanzadas durante el desarrollo de la práctica así como unas serie de posibles **líneas futuras**.

Además, se incluye un apéndice "***Contenidos del fichero entregable***" donde se desglosan las carpetas y ficheros disponibles en el entregable junto a esta memoria.

Capítulo 2

Selección de los artículos

En este capítulo se discutirá la selección de artículos que se ha realizado para la creación del clasificador de documentos.

Concretamente, se discutirá primero la **metodología** para la selección de estos artículos (incluyendo las fuentes de las que se extraerán). Tras esto se presentarán todos los **artículos elegidos**, junto a un enlace hacia la página web en la que se encuentran disponibles.

2.1. Metodología para la selección de artículos

De cara a obtener artículos de temáticas **disjuntas** (con el fin de facilitar la tarea al clasificador de artículos) se han elegido tres **temáticas** para los artículos:

- **Deportes**.
- **Política**.
- **Salud**.

De cada una de estas temáticas se elegirá un total de **30** artículos de fuentes diversas, cada uno de estos artículos cumpliendo las siguientes condiciones:

- El artículo está escrito en **español**.
- El artículo proviene de un **periódico digital**.
- La longitud del artículo es superior a **400 palabras**.
- Si un artículo fuese excesivamente largo (más de 1000 palabras), será **recortado**.
- Dentro de lo posible, se busca aportar un poco de **variedad** a cada temática en vez de centrarse únicamente en un solo tema.
- Se procurará que de los 30 artículos, los temas tratados estén repartidos de forma equitativa entre las dos mitades (primeros y últimos 15 artículos). Esto se hará para evitar sesgos posteriormente a la hora de la clasificación.

Los artículos seleccionados se guardarán en formato *TXT*, siendo su nombre $Articulo < Numero > - < Tematica >$. Por ejemplo, el tercer artículo de política tendrá el nombre *Articulo 03 - Política*. Cada fichero contiene en su interior el **título** del artículo, su **cabecera** (si la tiene) y el **cuerpo** del artículo.

2.2. Artículos seleccionados

Los artículos seleccionados se encuentran disponibles en la carpeta *Articulos*, dentro de la subcarpeta adecuada a su temática. Los artículos se encuentran ordenados de forma arbitraria (es decir, no están ordenados por grado de pertenencia a la temática, se asume que todos los artículos tienen el mismo grado de pertenencia)

A continuación, se mostrarán los artículos seleccionados para cada temática. El enlace a cada uno de los artículos está disponible a través de un **hipervínculo**, pulsando sobre cada título en las listas.

2.2.1. Deporte

Los artículos elegidos de **deportes** hablan principalmente sobre **futbol** y **baloncesto**, si bien incluyen también otros temas (como pueden ser los **Juegos Olímpicos** o los **Juegos Paralímpicos**).

Los artículos elegidos provienen de los siguientes periódicos:

- **El País**: `https://elpais.com/`.
- **As**: `https://as.com/`.
- **El Mundo**: `https://www.elmundo.es`.
- **La Vanguardia**: `https://www.lavanguardia.com/`.

Los treinta artículos seleccionados son los siguientes:

1. Lewandowski, mejor jugador para la FIFA.
2. La Premier vota contra las cinco sustituciones por partido, salvo en casos de golpes en la cabeza.
3. Real Sociedad-Barça y Real Madrid-Athletic, semifinales de la Supercopa de España.
4. Firmino arrebata el liderato al Tottenham en el último minuto.
5. El Valencia evita el desastre en la Copa.
6. James Harden, un problema de peso en Houston.
7. El Supremo de EE UU estudiará el caso de los deportistas universitarios que buscan cobrar un salario.
8. Así derrotó la NBA a Donald Trump.
9. Las seis superestrellas que regresan con mucho que demostrar.
10. Las lesiones se ceban otra vez con Gordon Hayward: fractura de dedo tras fichar por Charlotte.
11. Davis, LeBron y la 'llave Kuzma': la nueva dinastía de los Lakers.

12. De Kareem a Giannis: el hijo pródigo y el legado de los Bucks.
13. El TAS mantiene a Rusia fuera de los Juegos de Tokio 2021.
14. El COI, preparado para unos "Juegos seguros" en verano.
15. El deporte para romper barreras.
16. Otro lío con el VAR en Ipurua: "Creo que a Sergio Ramos también le ha parecido penalti".
17. Modric baila sobre la hoguera de Ipurua.
18. Florentino Pérez se queja del VAR y Tebas le responde: "Hay una opción para escuchar el partido con Real Madrid TV".
19. El Levante descuelga a la Real Sociedad.
20. 643, una cifra para la historia: Messi iguala a Pelé como máximo goleador de la historia de un club.
21. Luis Suárez echa abajo el muro y Costa lo remata.
22. Campazzo, Randolph, el enésimo desafío de Pablo Laso y la opción Ayón.
23. Jasikevicius pone a Heurtel en la calle.
24. El Barcelona conquista la Copa del Rey y cierra el año con 'triplete'.
25. Lucy Bronze gana el The Best FIFA Award.
26. El Maccabi ahoga al Barça y conquista el Palau.
27. Marc Gasol estrena conexión con Lebron James y Anthony Davis.
28. El Barça gana a la Penya sin Heurtel, a punto de dejar el club.
29. Un diamante de altura.
30. Paco Cubelos, sorprendido por los cambios de París 2024.

2.2.2. Política

Los artículos elegidos sobre **política** tratan temas generales de política (como puede ser **corrupción** o **temas de actualidad**) tanto a nivel **nacional** como **internacional**.

Los artículos elegidos provienen de los siguientes periódicos:

- **El Periódico**: `https://www.elperiodico.com`.
- **La Vanguardia**: `https://www.lavanguardia.com`.
- **El Mundo**: `https://www.elmundo.es/espana/2020/12/17/5fdb29d9fdddff637a8b4673.html`.
- **El Economista**: `https://www.eleconomista.es`.

- **The New York Times**: `https://www.nytimes.com/es`.
- **El Diario**: `https://www.eldiario.es/`.

Los treinta artículos seleccionados son los siguientes:

1. Juan Carlos I no volverá a España para pasar las fiestas de Navidad.
2. ERC plantará a JxCat en la constitución del Consell per la República.
3. La Audiencia Nacional saca a Villar Mir del 'caso Púnica': solo hay sospechas.
4. El rey Juan Carlos regulariza 678.393 euros ante las autoridades tributarias.
5. IU amplía su querella contra Juan Carlos I ante el Supremo.
6. Driss Oukabir, el "infiel".
7. El PSOE veta las 646 enmiendas a la Ley Celaá en el Senado para que no regrese al Congreso y se apruebe antes de Navidad.
8. La CE y representantes europeos se oponen a un acuerdo con Reino Unido de último minuto.
9. Pedro Duque se queda sin dirigir la Agencia Espacial Europea.
10. Montero abronca a Iglesias en los pasillos del Congreso y le llama "cabezón" tras las últimas discrepancias de Podemos con el PSOE.
11. El Govern cerrará 2020 con 5.000 millones más de gasto presupuestario.
12. Un acuerdo que Trump firmó dice que no puede vivir en Mar-a-Lago.
13. Biden y las oportunidades para Estados Unidos y América Latina.
14. Llamamos a los funcionarios de todos los estados. Ninguno tenía pruebas de fraude electoral significativo.
15. El epitafio de la independencia de Cataluña.
16. El primer médico español condenado por eutanasia: "Soy activista de la vida, pero sufrir ante lo inevitable es estúpido".
17. Los ultras de HazteOir logran apoderarse de las protestas contra la ley Celaá y la eutanasia.
18. ¿Qué pasará con la inmersión lingüística? Entre la sentencia que la tumba y la esperanza en la 'ley Celaá'.
19. Católicos de Toledo cargan contra la nueva ley LGTBI de Castilla-La Mancha porque impide un "tratamiento médico" de la homosexualidad.
20. El Ayuntamiento de Madrid elimina las ayudas directas a organizaciones históricas por los derechos LGTBI.
21. La Fiscalía comprueba si el pago de 678.000 euros de Juan Carlos I a la Comunidad de Madrid en impuesto de donaciones cubre su deuda fiscal.

22. España se coloca ya a las puertas de reconocer el derecho a la eutanasia.
23. El Gobierno se apoya en la gran empresa para gastar los fondos europeos.
24. El Gobierno acusa al CGPJ de "invadir la soberanía parlamentaria".
25. Pedro Sánchez llama a Pablo Casado pero no avanzan en la renovación del CGPJ.
26. Londres y Bruselas extienden las negociaciones del Brexit.
27. Trump contradice a su secretario de Estado y cuestiona que Rusia estuviera detrás del ciberataque masivo que ha sufrido EEUU.
28. EEUU sufre el peor 'hackeo' de su historia, con ciberataques que abren brechas de seguridad en Defensa, Comercio y Energía.
29. Michel Barnier avisa de que sólo quedan "algunas horas" para tener el pacto posBrexit.
30. Vladimir Putin afirma que presentar a Rusia como agresiva "es tomarnos por imbéciles".

2.2.3. Salud

Los artículos elegidos de **salud** tratan principalmente sobre la **pandemia del COVID-19**, como era de esperar por la situación actual. Ahora bien, se ha procurado también tratar algunos otros temas médicos. Además, se han elegido noticias relacionadas con **España** y con **el resto del mundo**.

Los artículos elegidos provienen de los siguientes periódicos:

- **El Español**: `https://www.elespanol.com`.
- **El Tiempo**: `https://www.eltiempo.com/`.
- **El Diario**: `https://www.eldiario.es`.
- **El Mundo**: `https://www.elmundo.es/`.
- **La Razón**: `https://www.larazon.es/`.
- **La Vanguardia**: `https://www.lavanguardia.com`.

Los treinta artículos seleccionados son los siguientes:

1. El llamamiento desesperado de médicos a pacientes: "Los quirófanos son seguros".
2. La Covid se ensaña con Canarias, el destino navideño preferido de España.
3. La telemedicina hace más partícipe al paciente en su salud.
4. ¿Ibuprofeno o paracetamol? Éste es el mejor medicamento para la fiebre de la Covid.
5. Madrid afronta la Navidad en plena subida de casos y con sus peores cifras en casi un mes.
6. EPS debe cubrir cirugía de abdominoplastia tras un bypass.

7. México supera las 114 mil muertes por Covid-19.
8. Lo que se sabe de la nueva variante del Sars-CoV-2 en el Reino Unido.
9. Doctora tuvo que aprender a hablar tras contagiarse con coronavirus.
10. Cantante de RBD se defiende de hombre que lo acusó de transmitirle VIH.
11. ¿Estar encerrado aumenta el consumo excesivo de alcohol?
12. William Shakespeare, la segunda persona vacunada contra la covid-19.
13. El coronavirus, en datos: mapas y gráficos de la evolución de los casos en España y el mundo.
14. De pasar 25 días en coma a volver a competir.
15. Cómo cuidar el corazón en tiempos de COVID.
16. La nueva cepa del coronavirus, más contagiosa y dominante.
17. Repunta el coronavirus: la incidencia vuelve a aumentar y llega a 214 casos por 100.000 habitantes.
18. El Centro de Investigación Príncipe Felipe desarrolla un test para la Covid-19 a partir de muestras de saliva
19. Los pacientes Lázaro: el 'milagro' de sobrevivir al cáncer más letal ya tiene explicación.
20. La Agencia del Medicamento de EEUU recomienda aprobar la vacuna de Moderna contra el Covid.
21. El Tribunal Supremo de Brasil avala que la vacuna contra la covid-19 sea obligatoria.
22. Una sanitaria de EEUU sufre una fuerte reacción alérgica a la vacuna contra el coronavirus de Pfizer.
23. La OMS pide que se use mascarilla en las reuniones familiares de Navidad en Europa.
24. Sin tratamiento el 40% de las personas con riesgo de fractura.
25. Confirman que la Covid multiplica por cinco el riesgo de morir en comparación con la gripe.
26. La OMS pide «cuadruplicar» los esfuerzos contra el ébola.
27. Reducir la inflamación en el riñón para frenar el envejecimiento.
28. Pediatras aconsejan vacunación universal de la gripe para mayores de 6 meses.
29. Los riesgos de una epidemia que cambia de nombre.
30. La nueva cepa del coronavirus que inquieta a todos los epidemiólogos.

Capítulo 3

Extracción de glosarios

En este capítulo se presentará el proceso realizado para la extracción de glosarios para cada una de las temáticas.

Primero se describirá la metodología seguida para la extracción (centrandose en el **preprocesamiento de los textos** y la **selección y variación morfológica** realizada), tras lo cual se comentará brevemente la implementación del extractor y finalmente se mostrarán los glosarios obtenidos.

3.1. Metodología para la extracción de glosarios

Para realizar la **extracción de glosarios** se seguirá una metodología, de cara a hacer el proceso reproducible. La metodología seguida es la siguiente:

1. Antes de comenzar con el proceso, es necesario tener un **conjunto de textos** para cada temática.

   Estos textos se usarán para extraer el glosario de ellos, y se tomarán como referentes. Se tomarán los **15 primeros artículos** (artículos del 1 al 15) de cada temática para esto.

2. Se carga una lista de *stopwords*.

   Esta lista será usada posteriormente en el paso de preprocesamiento para eliminar palabras inútiles, y se encuentra en el fichero *stopwords.txt*. En caso de que no estuviese el fichero disponible, usaría la lista de palabras por defecto contenida en la librería *NLTK*.

   El fichero *stopwords.txt* provisto proviene originalmente del proyecto **Stopwords ISO** (disponible la versión en español en `https://github.com/stopwords-iso/stopwords-es`). A este fichero original posteriormente se le añadieron algunas palabras adicionales que se consideraron irrelevantes.

3. Se cargan todos los **textos** a usar para la extracción de glosarios de los ficheros en los que se almacenan, **preprocesando** cada texto para reducir la carga de trabajo del extractor y para eliminar palabras irrelevantes.

El proceso exacto de preprocesamiento será descrito posteriormente.

4. Una vez los textos están cargados y preprocesados, se identificarán las **60 palabras más frecuentes** dentro de cada temática y, de cada glosario inicial, se extraen manualmente **20 palabras**.

   Se ha optado por usar la **frecuencia de las palabras** como la métrica para elegir las palabras más relevantes, considerándose que las palabras que aparecen muchas veces en un temario son palabras **identificativas**. Si bien existen métricas más elaboradas, se ha optado por esta debido a su simplicidad y a que sus carencias serán suplidas posteriormente de forma manual. Se ha usado la clase *Counter* para realizar este cálculo de la frecuencia.

   Se identifica un número superior a las palabras finales a elegir en el glosario (**20**) para poder cribar palabras que no sean útiles (ya sea por estar repetidas, por no ser consideradas relevantes...) El procedimiento exacto para elegir las 20 palabras será descrito posteriormente.

5. Ya con las **20 palabras** de cada temática elegidas, se procederá a realizar la **variación morfológica** de cada termino, para obtener el **glosario final**.

Por tanto, la **extracción de glosarios** se realiza de forma semi-automática, realizando el programa un cribado y selección inicial de términos que posteriormente será refinada y expandida manualmente.

3.1.1. Preprocesamiento de los textos

De cara a preprocesar los textos, se llevarán a cabo los siguientes pasos:

1. Se *tokeniza* el texto para trabajar con él.

   Tokenizar el texto consiste en dividirlo en una lista de palabras, de forma que sea posible trabajar con cada palabra por separado. En este caso es de interés para poder procesar cada palabra por separado y para, posteriormente, poder contar la frecuencia de cada palabra de forma más sencilla.

   Esta *tokenización* se lleva a cabo mediante *NLTK*. El kit de herramientas de lenguaje natural, o NLTK, es un conjunto de bibliotecas y programas para el procesamiento del lenguaje natural (PLN) en el lenguaje de programación Python.

2. Se pasa el texto a **minúsculas**.

   Las **mayúsculas** no aportan ninguna información relevante y solo pueden provocar que palabras iguales se identifiquen como distintas, por lo que se eliminan.

3. Se eliminan **los caracteres que no aportan información semántica al texto** (puntos, interrogaciones comillas...).

   Estos caracteres no aportan ninguna información que pueda ser relevante para los glosarios o los clasificadores, por lo que son eliminados.

4. Se eliminan las "**stopwords**".

   Una "*stopword*" (o **palabra de parada**) es una palabra en cualquier idioma que no añade significado importante a una frase. Al no aportar información util, pueden ser ignoradas con seguridad sin sacrificar el significado de la frase. Algunos ejemplos de estas palabras en español son *0*, *1*, *él*, *actualmente*, *ahí*...

En este caso, **no se realizará una reconstrucción del texto** a partir de los tokens. Esto se debe a que los tokens facilitarán la tarea de calculo de las palabras de máxima frecuencia de aparición.

Se consideró realizar también **stemming** (reducción de las palabras a su raíz) de cara a reducir la variabilidad (actualmente, *jugador* y *jugadora* se consideran palabras distintas). Ahora bien, por recomendación del profesor y para ceñirse mejor al enunciado de la práctica (siendo necesario crear un **glosario de términos**), se optó por no utilizar esta técnica.

Para mostrar un ejemplo de **preprocesamiento**, se muestra a continuación un fragmento del *Artículo 1 de Deportes*:

> Lewandowski, mejor jugador para la FIFA
>
> El organismo rector del fútbol premia con The Best al goleador del Bayern, campeón de Bundesliga, Copa, Champions y Supercopas de Europa y Alemania con cifras de récord en el año de la pandemia.
>
> Las cifras representan la identidad del goleador: 9, 30, 36, 28, 25, 42, 43, 41, 40 y 55. Son los goles totales que marcó Robert Lewandowski en la Bundesliga cada una de las diez temporadas sucesivas desde que fichó por el Borussia Dortmund en 2010 y pasó al Bayern Múnich en 2014. Contra el orden de la fuerza biológica, que comienza a perder camino de sus 33 años, y contra la lógica del individualismo, el futbolista alcanzó el pico de su producción cuando hizo un esfuerzo adicional por ayudar a sus compañeros. Como dijo Lothar Matthäus tras observar sus dos goles al Wolfsburgo, el miércoles: "¡Ahora es menos egoísta!".

Para comparar, la versión preprocesada del mismo fragmento es la siguiente:

> lewandowski mejor jugador fifa organismo rector fútbol premia the best goleador bayern campeón bundesliga copa champions supercopas europa alemania cifras récord año pandemia cifras representan identidad goleador goles totales marcó robert lewandowski bundesliga cada diez temporadas sucesivas fichó borussia dortmund pasó bayern múnich orden fuerza biológica comienza perder camino años lógica individualismo futbolista alcanzó pico producción hizo esfuerzo adicional ayudar compañeros dijo lothar matthäus tras observar dos goles wolfsburgo miércoles menos egoísta

Como se puede observar se ha eliminado gran parte de la información superflua, lo que facilitará el trabajo posteriormente de cara a extraer el glosario. Aun así será necesario realizar un cribado manual para suplir las faltas del procesado automático.

3.1.2. Creación de glosarios

Para obtener los glosarios finales, es necesario refinar los glosarios propuestos por la aplicación mediante la **elección de 20 palabras** y la **variación morfológica**.

Para seleccionar las **20 palabras más relevantes** de la lista de 60 palabras más frecuentes, se han seguido las siguientes indicaciones:

- Las palabras se consideran en orden de **mayor a menor frecuencia**.
- Las palabras **no deben repetirse** ni entre glosarios ni dentro del propio glosario.

  Si, por ejemplo, se ha seleccionado **jugador** para el glosario y posteriormente aparece **jugadores**, se debe descartar la segunda palabra para evitar repeticiones. La idea es que el glosario esté formado por términos únicos y útiles para discriminar temáticas.
- Las palabras deben ser, a ser posible, **sustantivos** o **siglas**.

  Se ha considerado que los **sustantivos** son las palabras que más identificativas van a poder resultar en el glosario. Por ejemplo, los **verbos** tienen demasiadas formas verbales (que nuestros clasificadores identificarían como palabras distintas) como para ser identificativos.

  Respecto a **siglas**, esto se ha decidido tras considerar que algunos **nombres propios** (como, por ejemplo, Trump o Lewandosky) serían demasiado específicos para pertenecer al glosario, pero otras siglas (como *CUP*) sí que son suficientemente relevantes como para que sea interesante considerarlas.
- Las palabras deben ser, a ser posible, **significativas**.

  Esta parte del estudio es la más subjetiva. A la hora de cribar la lista, es posible que alguna palabra acabe descartada por no ser considerada realmente significativa o haber otras palabras en la lista más importantes.

Una vez seleccionadas las 20 palabras para el glosario de cada temática, se procede a la **variación morfológica**. Para cada palabra elegida en el glosario, se añadirán al glosario también todas sus otras formas. Por ejemplo, si se selecciona **jugador** para el glosario de Deportes, se añadirán al glosario las palabras *jugador, jugadora, jugadores* y *jugadoras*.

Esto se hace de esta forma para cubrir todas las posibilidades de cada palabra. Si la palabra **jugador** se considera importante, tiene sentido considerar que la palabra **jugadora** también lo sea. Por tanto, los glosarios finales realmente tendrán más de 20 palabras.

3.2. Implementación y uso del extractor de glosarios

En esta sección se comentará la implementación y el uso del extractor de glosarios, tanto el **lenguaje y las dependencias** requeridas como la **forma de uso** del programa implementado.

3.2.1. Implementación y dependencias

El extractor de glosarios está disponible en el fichero *extractor_terminologico.py*. Este programa ha sido implementado en **Python 3.7.6**, y utiliza las siguientes librerías:

- *NLTK (Natural Language ToolKit)* v3.5

La librería *NLTK* requiere unos pasos adicionales para su instalación (para descargar los *dataset* y modelos que utiliza), siendo necesarios concretamente los siguientes comandos para su instalación:

```
pip install nltk
python -m nltk.downloader all
```

El programa ha sido probado únicamente con las versiones descritas. Si bien es posible que funcione con otras versiones de Python o de las librerías, no se puede garantizar su compatibilidad.

La implementación como tal se encuentra documentada en detalle en el fichero indicado previamente, por lo que se recomienda encarecidamente su lectura para observar los detalles técnicos.

3.2.2. Uso del programa

El programa se utiliza mediante el siguiente comando:

```
python extractor_terminologico.py <ruta>
```

Donde el argumento *ruta* es una **ruta** (relativa o absoluta) a una carpeta conteniendo los ficheros a usar para generar el glosario. Concretamente, la carpeta *ruta* debe contener a su vez tantas **subcarpetas** como temáticas se quieran extraer, teniendo cada subcarpeta como nombre el nombre de la temática y en su interior los documentos en *TXT* tal y como se describieron en el Capítulo 2.

Los resultados del programa se imprimirán en pantalla además de ser escritos en un fichero *glosarios.txt*, donde se encontrarán disponibles los glosarios iniciales (sin procesar) de cada una de las temáticas.

En concreto, para usar el programa con los ficheros adjuntos a la memoria, el comando a utilizar es:

```
python extractor_terminologico.py Glosario
```

Donde **Glosario** es una carpeta conteniendo los **15 primeros artículos** (artículos del 1 al 15) de cada temática. Esta carpeta será reutilizada en el clasificador para entrenar los modelos.

3.3. Glosarios obtenidos

En esta sección se mostraran los glosarios obtenidos, tanto los **glosarios iniciales** devueltos directamente por la herramienta como los **glosarios procesados** manualmente y que se usarán definitivamente en el clasificador de documentos.

3.3.1. Glosarios iniciales

Para cada temática se muestra el glosario inicial de **50 palabras** devuelto por el extractor de glosarios. Este glosario será expresado en forma de lista, donde cada palabra va acompañada de su **frecuencia total** en los artículos de esa temática usados para la creación del glosario. Además, las palabras **seleccionadas para procesar en el glosario** están resaltadas en amarillo.

Glosario de deportes

temporada - (33), equipo - (29), lakers - (28), nba - (24), jugadores - (22), millones - (22), lebron - (20), franquicia - (20), jugador - (19), contrato - (19), mercado - (17), partido - (16), juegos - (16), harden - (15), historia - (14), estrella - (14), lewandowski - (12), fútbol - (12), pandemia - (12), temporadas - (12), equipos - (12), liga - (12), durant - (12), bucks - (12), campeón - (11), liverpool - (11), real - (11), james - (11), lesión - (11), mundo - (11), ama - (11), bayern - (10), asamblea - (10), presidente - (10), nivel - (10), ganar - (10), copa - (9), madrid - (9), partidos - (9), firmino - (9), tottenham - (9), cantidad - (9), playoffs - (9), ncaa - (9), rusia - (9), olímpicos - (9), panam - (9), sports - (9), competición - (8), verano - (8)

Glosario de política

gobierno - (26), política - (25), presidente - (23), trump - (22), juan - (20), carlos - (20), rey - (20), erc - (17), españa - (15), emérito - (15), driss - (13), pandemia - (12), portavoz - (12), investigación - (12), euros - (11), izquierda - (11), catalana - (11), unidos - (10), tribunal - (10), miembros - (10), aguas - (10), director - (10), aschbacher - (10), club - (10), jueves - (9), exterior - (9), grupo - (9), consejo - (9), agencia - (9), atentados - (9), psoe - (9), biden - (9), país - (8), carta - (8), puigdemont - (8), elecciones - (8), caso - (8), relación - (8), cargo - (8), querella - (8), congreso - (8), unidas - (8), reino - (8), unido - (8), ue - (8), govern - (8), ciudad - (8), independencia - (8), casa - (7), jxcat - (7)

Glosario de salud

casos - (60), salud - (45), pacientes - (38), pandemia - (35), país - (34), virus - (26), personas - (23), coronavirus - (22), número - (21), caso - (21), dolor - (19), españa - (19), datos - (19), paciente - (18), cirugía - (17), mundo - (17), situación - (17), canarias - (16), vida - (16), incidencia - (16), enfermedad - (16), hospital - (15), hospitales - (14), madrid - (14), covid - (14), países - (14), méxico - (14), vih - (14), chávez - (14), forma - (13), millones - (13), comunidad - (13), vacuna - (13), mil - (13), habitantes - (12), muertes - (12), semanas - (11), mayo - (11), riesgo - (11), problema - (11), diciembre - (11), casa - (11), ibuprofeno - (11), octubre - (11), seguridad - (10), población - (10), semana - (10), control - (10), sanidad - (10), eps - (10),

3.3.2. Glosarios procesados

En la tabla 3.1 se muestran los tres glosarios finales (procesados) que se utilizarán con el clasificador de documentos. Estos glosarios se han obtenido mediante **variación morfológica** sobre los 20 términos elegidos de los glosarios iniciales, añadiendo todas las variantes de cada uno de estos términos.

Índice	Deporte	Política	Salud
1	Temporada/Temporadas	Gobierno/Gobiernos	Caso/Casos
2	Equipo/Equipos	Política/Políticas	Salud
3	NBA	Presidente/Presidenta/ Presidentes/Presidentas	Paciente/Pacientes
4	Jugador/Jugadora/ Jugadores/Jugadoras	Rey/Reina/ Reyes/Reinas	Pandemia/Pandemias
5	Millon/Millones	ERC	País/Países
6	Franquicia/Franquicias	España	Virus/Viruses
7	Contrato/Contratos	Emérito/Emérita/ Eméritos/Eméritas	Persona/Personas
8	Mercado/Mercados	Portavoz/Portavoces	Coronavirus
9	Partido/Partidos	Investigación/ Investigaciones	Número/Números
10	Juego/Juegos	Euro/Euros	Dolor/Dolores
11	Historia/Historias	Izquierda/Izquierdas	Dato/Datos
12	Estrella/Estrellas	Catalán/Catalana/ Catalanes/Catalanas	Cirugía/Cirugías
13	Fútbol	Tribunal/Tribunales	Mundo/Mundos
14	Olímpico/Olímpica/ Olímpicos/Olímpicas	Miembro/Miembros	Situación/Situaciones
15	Liga/Ligas	Agua/Aguas	Vida/Vidas
16	Campeón/Campeona/ Campeones/Campeonas	Director/Directora/ Directores/Directoras	Incidencia/Incidencias
17	Lesión/Lesiones	Club/Clubes	Enfermedad/Enfermedades
18	Asamblea/Asambleas	Exterior/Exteriores	Hospital/Hospitales
19	Nivel/Niveles	Grupo/Grupos	COVID
20	Copa/Copas	Consejo/Consejos	Vacuna/Vacunas

Cuadro 3.1: Glosarios procesados de las tres temáticas

Capítulo 4

Clasificador documental

En este capítulo se presentará el proceso realizado para la **clasificación de documentos**, asignándole a cada documento una clase entre las temáticas propuestas.

Primero se presentarán una serie de **conceptos** usados por el clasificador de documentos (concretamente las **métricas** y **modelos** que se utilizarán). Tras esto, se presentará la **metodología** seguida por el desarrollo y la evaluación de los clasificadores. Finalmente, se comentará brevemente la **implementación** y se expondrá un manual de usuario del programa.

4.1. Conceptos utilizados por el clasificador de documentos

Antes de explicar la metodología que se ha seguido para la clasificación documental, se detallarán las **métricas** y **modelos** que se probarán. A cada combinación de métrica y modelo se le denominará como **clasificador**, existiendo un total de **6 clasificadores** a probar.

4.1.1. Métricas de clasificación utilizadas

En este contexto, se entenderá como **métrica** al conjunto de valores que se usarán para **calcular la distancia entre documentos**. Concretamente se estudiarán dos métricas: la *frecuencia absoluta* y *TF-IDF*.

Frecuencia absoluta de cada término

La **frecuencia absoluta de cada termino** es una de las métricas más simples existentes y siendo la variante de *bag-of-words* más sencilla existente. Consiste simplemente en calcular la frecuencia absoluta (el número de veces que aparece cada termino) de cada palabra del glosario en cada documento. Si bien es una métrica muy simple, es interesante probarla y usarla como *baseline* para contrastarla contra TF-IDF.

Para usar esta métrica se utilizará la clase *CountVectorizer* de la librería *scikit-learn*.

TF-IDF

TF-IDF (también conocida como *Term Frequency times Inverse Document Frequency*, que podríamos traducir como *frecuencia del término por frecuencia inversa del documento*) es una

métrica *bag-of-words* muy popular en los campos de procesamiento de lenguaje natural. Se usa para encontrar los términos más **relevantes** de cada documento dentro de un *corpus* de documentos.

Su funcionamiento se basa en medir **con qué frecuencia aparece un término o frase dentro de un documento** determinado, y **comparar esta medida con el número de documentos que mencionan ese término** dentro de un corpus de documentos. De esta forma, se da más valor a los términos muy característicos de un documento frente a terminos que aparecen mucho pero en todos los documentos (como pueden ser, por ejemplo, las *stopwords*)

El cálculo de esta métrica se divide en dos partes:

- **Frecuencia del término** ($TF(t, d)$): Cuenta la frecuencia (el numero de veces que aparece) de un término t en un documento d. Esta frecuencia puede estar ponderada por algún tipo de formula. se aplican una serie de modificadores a los términos. Por lo que la relevancia de un documento no se basa en la repetición de una palabra clave.
- **Frecuencia inversa de documento** ($IDF(t)$): Calcula, mediante una formula, la proporción de documentos del corpus en los que aparece el término t.

Existe una gran variedad de variantes de **TF-IDF**, por lo que las fórmulas usadas concretamente por este trabajo son:

$$tf(t, d) = \text{frecuencia del termino } t \text{ en el documento } d$$

$$idf(t) = log\frac{1 + n}{1 + df(t)} + 1,$$

donde n es el número de documentos en el corpus y $df(t)$ es el número de documentos que contienen el término t. Además, la fórmula está suavizada (se considera que siempre hay al menos un documento donde aparece cada termino) para evitar divisiones entre 0.

El cálculo final de la métrica para un documento y un término es el siguiente:

$$tf\text{-}idf(t, d) = tf(t, d) \cdot idf(t)$$

Para usar esta métrica se utilizará la clase *TfidfVectorizer* de la librería *scikit-learn*, con sus parámetros iniciales.

4.1.2. Modelos de clasificador utilizados

En total se van a probar tres modelos distintos con las métricas descritas: **kNN**, **Naive Bayes** y **máquinas de vector de soporte**. Para cada modelo se describirá brevemente su funcionamiento y se describirán los parámetros a ajustar de cada modelo.

K vecinos más cercanos (kNN)

El modelo de **K vecinos más cercanos** (también conocido como *K-Nearest Neighbours*) es un modelo de clasificación simple, que consiste en una base de datos de casos, cada caso con sus características y su clasificación.

Cuando llega un caso nuevo, será clasificado con la clase más frecuente entre los k vecinos más cercanos de la base de datos (midiendose la cercanía mediante una función de distancia).

La implementación usada es la de *KNeighborsClassifier*, perteneciente a la librería *scikit-learn*. Esta clase utiliza la **distancia euclidiana** como función de distancia. Los parámetros que se ajustarán para cada métrica son los siguientes:

- **Número de vecinos**: Se probará el rendimiento considerando valores para k entre 1 y 10.
- **Ponderación del peso de los vecinos**: Se probarán dos ponderaciones a la hora de elegir el peso que tiene cada vecino en la clasificación:
  - *Uniforme*: Todos los vecinos tienen el mismo peso.
  - *Por distancia*: Los vecinos a mayor distancia influyen menos en la clasificación final que los más cercanos.

Naive Bayes

Naive Bayes es una familia de simples clasificadores probabilísticos basados en la aplicación del **Teorema de Bayes** con fuertes suposiciones de independencia entre las características. En otras palabras, se asume que la presencia de una cierta característica en un conjunto de datos no está en absoluto relacionada con la presencia de cualquier otra característica.

Naive Bayes se encuentra entre los modelos de redes bayesiana **más simples**, pero debido a su simplicidad y a su buen rendimiento en muchos campos (entre los que se incluye, notablemente, la clasificación documental), siguen siendo modelos muy relevantes.

La implementación usada es, concretamente, la de *MultinomialNB* de la librería *scikit-learn*. Si bien existen otras implementaciones más típicas (como *GaussianNB*), se utiliza esta al ser **capaz de trabajar con números enteros y reales** y al ser recomendada para clasificación documental. No se ajustará ningún parámetro para este modelo.

Máquina de vector de soporte (SVM)

Las **máquinas de vector de soporte** (también conocidas como *SVM* de sus siglas en inglés *Support Vector Machine*) son modelos de aprendizaje supervisado basados en la teoría de Vapnik–Chervonenkis con algoritmos especializados en problemas de **clasificación** (nuestro problema) y regresión.

La idea simplificada del modelo es, dado un conjunto de datos dividido en **dos** clases, encontrar el **hiperplano** (de cualquier dimensionalidad) que separe los elementos de las dos clases con el mayor margen posible. Una vez encontrado dicho hiperplano, los nuevos casos se clasificarán dependiendo del lado del hiperplano en el que queden.

La implementación usada es la clase *SVC* de la librería *scikit-learn*. Normalmente las máquinas de vector de soporte sirven únicamente para distinguir entre dos clases, pero esta implementación ofrece la posibilidad de clasificación multiclase de forma *uno contra uno* (se crea un clasificador para todos los pares de características). Para este modelo se ajustará un único parámetro:

- **Tipo de kernel**: Esto indica el tipo de kernel (separación) que se realizará para dividir el conjunto de datos en dos clases. Existen varias formulas posibles:
  - Lineal.
  - Polinomial. En este caso, se probarán grados del polinomio entre 2 y 5.
  - RBF (*Radial Basis Function*).
  - Sigmoide.

4.2. Metodología para la clasificación de los documentos

De nuevo, para realizar la **clasificación de documentos** se seguirá una metodología de cara a hacer el proceso reproducible. La metodología seguida concretamente es la siguiente:

1. Antes de comenzar con el proceso, es necesario contar con **dos conjuntos de textos**: uno para **entrenar los clasificadores** (a ser posible, el mismo conjunto usado para extraer el glosario) y otro para **ser clasificado** (de cara a probar el rendimiento del clasificador).

   Para obtener estos dos conjuntos se separará el conjunto de artículos en dos: los **15 primeros artículos** (artículos de 1 a 15) de cada temática se usarán para entrenar a los modelos y los **15 últimos artículos** (artículos de 16 a 30) serán clasificados.

2. Se carga una lista de *stopwords*, se cargan los textos a partir de los ficheros seleccionados y se preprocesan todos los textos.

   El preprocesamiento realizado es idéntico al presentado en la extracción de glosarios, con la única diferencia de que los textos preprocesados serán **reconstruidos** (devueltos en forma de un único texto) en vez de quedarse tokenizados. Esto es necesario para poder trabajar posteriormente con ellos.

3. Se genera un **glosario** general a partir de los tres glosarios generados.

   Este glosario se utilizara para, a la hora de aplicar las métricas, tener en cuenta únicamente las palabras relevantes y poder ignorar el resto de palabras. De esta forma se reduce la complejidad del problema.

   La razón por la que se usa el mismo conjunto de textos para extraer el glosario y para entrenar los clasificadores es esta: ya que los glosarios se fusionan, entrenar los clasificadores con el mismo conjunto de textos garantiza que asociará a cada temática las palabras de su glosario apropiado (al ser más frecuentes en esos textos).

4. Se **aplican las métricas** (*frecuencia absoluta* y *TF-IDF*) a cada uno de los conjuntos de textos procesados, obteniendo representaciones vectorizadas de cada conjunto.

   Esta transformación devuelve representaciones simplificadas de los documentos, y serán utilizadas como entradas de los clasificadores (tanto para entrenarlos como para clasificarlos

posteriormente). La transformación se hace **teniendo en cuenta el glosario** (solo se almacena la métrica para las palabras del glosario).

Las clases usadas por cada métrica han sido descritas en la sección anterior.

5. Para cada par de métrica y modelo, se **entrena al clasificador** usando la representación vectorizada del primer conjunto de textos.

   A la hora de entrenar los modelos se utilizará un proceso de **GridSearch** para optimizar el modelo, encontrando la combinación de parámetros óptima. Este consiste en probar el rendimiento del modelo con todas las combinaciones de parámetros propuestas, devolviendo la que devuelve la tasa de acierto más alta. Esto se ha realizado usando la clase *GridSearchCV*, parte de la librería *scikit-learn*.

   Los parámetros a probar de cada modelo y las implementaciones específicas para cada modelos se han descrito en la sección anterior.

6. Para cada par de métrica y modelo entrenado, se **clasifican** los textos del segundo conjunto de textos, obteniendo la **tasa de acierto** de cada clasificador.

   La métrica que se usará para medir el rendimiento del clasificador es la **tasa de acierto** (*accuracy*), y será usada para determinar el mejor clasificador de todos los estudiados. Además de la tasa de acierto general, se calculará la tasa de acierto por separado para cada temática.

7. Para cada par de métrica y modelo entrenado, se **ordenarán** los textos clasificados.

   Esta ordenación se realizará para cada temática, usando todos los textos clasificados como esa temática. La métrica usada para ordenar los textos es la **probabilidad de la clase** (la probabilidad de que el clasificador asigne la temática al texto) devuelta por los modelos, ordenandolos de mayor a menor probabilidad.

   Esta probabilidad es devuelta de forma nativa por kNN y Naive Bayes, pero para las máquinas de vector soporte es necesario simularla (usando un parámetro proporcionado por el modelo). Al ser simulada el modelo no garantiza que los valores sean consistentes, por lo que es posible que la ordenación varíe ligeramente entre ejecuciones.

8. Finalmente, se genera una serie de **gráficas** mostrando el rendimiento de los clasificadores.

   Se generarán en total cuatro gráficos de barras, uno para cada tasa de acierto (general y para cada temática). Estas gráficas se generan de cara a ser usadas en esta memoria, para poder comparar visualmente de forma sencilla el rendimiento de todos los clasificadores.

4.3. Implementación y uso del clasificador de documentos

En esta sección se comentará la implementación y el uso del clasificador de documentos, describiendo tanto **el lenguaje y las dependencias requeridas** como la **forma de uso** del programa implementado.

4.3.1. Implementación y dependencias

El clasificador de documentos está disponible en el fichero *clasificador_documental.py*. Este programa ha sido implementado en **Python 3.7.6**, y utiliza las siguientes librerías;

- *NLTK (Natural Language ToolKit)* v3.5
- *numpy* v1.18.1
- *matplotlib* v3.1.2
- *scikit-learn* v0.22.1

El programa ha sido probado únicamente con las versiones descritas. Si bien es posible que funcione con otras versiones de Python o de las librerías, no se puede garantizar su compatibilidad.

La implementación como tal se encuentra documentada en detalle en el fichero indicado previamente, por lo que se recomienda encarecidamente su lectura para observar los detalles técnicos.

4.3.2. Uso del programa

El programa se utiliza mediante el siguiente comando:

```
python clasificador_documental.py <ruta_entrenamiento> <ruta_test>
```

Donde:

- **ruta_entrenamiento** es una ruta (relativa o absoluta) a una carpeta conteniendo los ficheros a utilizar para generar el conjunto de entrenamiento de los modelos.
- **ruta_test** es una ruta (relativa o absoluta) a una carpeta conteniendo los ficheros a utilizar para generar el conjunto de test de los modelos, es decir, los documentos a clasificar.

Ambas carpetas deben contener, a su vez, tantas **subcarpetas** como temáticas se quieran extraer, teniendo cada subcarpeta como nombre el nombre de la temática y en su interior los documentos en formato *TXT* tal y como se describieron en el Capítulo 2. La razón por la que **ruta_test** debe seguir también esta estructura es porque se utilizarán las temáticas reales de los artículos para calcular la tasa de acierto de cada clasificador.

Todos los resultados del programa (tanto la tasa de acierto de cada clasificador como la ordenación de artículos propuesta) se imprimirán en pantalla y quedarán escritos en el fichero *clasificador_resultados.txt*. Además, para cada clasificador se creará una carpeta de nombre *Clasificacion_<nombre de la métrica>_<nombre del modelo>* donde se almacenarán copias de los ficheros, ordenadas de mayor a menor pertenencia a la temática.

Además, el programa genera **gráficas**, tanto en *PNG* como en *EPS*, que serán utilizadas para el análisis de los resultados en la memoria.

Para el uso del programa en esta circunstancia concreta, el comando a utilizar es:

```
python clasificador_documental.py Glosario Por_clasificar
```

Donde:

- **Glosario**: Carpeta que contiene los artículos usados para la extracción del glosario. Esta carpeta contiene los **15 primeros artículos** (artículos del 1 al 15) de cada temática.
- **Por_clasificar**: Carpeta que contiene los artículos usados para ser clasificados. Concretamente, contiene los **15 últimos artículos** (artículos del 16 al 30) de cada temática.

Capítulo 5

Experimentación

En este capítulo se detallará toda la **experimentación** realizada con el clasificador documental descrito en el capítulo anterior. Concretamente, se probarán todos los pares de **métrica** y **modelo** que se describieron, siguiendo el procedimiento.

Primero se plantearán una serie de **hipótesis** sobre los resultados que se esperan obtener de estos modelos. Tras esto, se exponen los **resultados** obtenidos, analizándolos y comparándolos entre ellos. Finalmente, se expondrán unas **conclusiones** respecto a las hipótesis planteadas.

5.1. Hipótesis planteadas

De cara a la experimentación realizada, hay una serie de **hipótesis** o resultados esperados con los que partimos:

- **Los glosarios obtenidos son representativos de cada temática:**

  Los **glosarios** que se han obtenido en el Capítulo 3 son totalmente disjuntos (no se repiten palabras entre ellos) y están formados por sustantivos muy característicos de cada campo.

  Por tanto, se puede esperar que usando dichos glosarios para distinguir las temáticas de los documentos se obtengan resultados buenos.

- **La métrica TF-IDF dará mejores resultados que la frecuencia absoluta:**

  La **frecuencia absoluta** es una de las métricas más simples que se pueden utilizar en clasificación de documentos, midiendo únicamente las veces que aparece cada palabra del glosario en cada documento. Una clasificación tan simple puede llevar a sesgos (como, por ejemplo, palabras que tienen más peso porque aparecen en documentos más largos).

  En cambio, **TF-IDF** (si bien sigue siendo una métrica de *bag of words* relativamente simple) ofrece un mayor grado de información (siendo capaz de distinguir mejor qué términos son relevantes y cuales no), por lo que es de esperar que obtenga resultados mejores.

- **Los modelos de Naive Bayes y de máquinas de vector de soporte darán mejores resultados que kNN:**

  Tanto **Naive Bayes** como las **máquinas de vector de soporte** son modelos típicamente usados para clasificación de documentos, debido a su buen rendimiento con *datasets* de alta dimensionalidad (muchas características), como pueden ser los textos procesados para clasificación de documentos.

  Por tanto, se puede esperar que estos modelos obtengan mejores resultados que **kNN** (un modelo más genérico).

- **El modelo que mejores resultados ofrecerá es la máquina de vector de soporte:**

  Naive Bayes es un modelo que considera todas las características de la entrada independientes entre ellas, mientras que las **máquinas de vector de soporte** tienen en cuenta las posibles relaciones que existen entre características.

  Por tanto, en clasificación de documentos (donde existen relaciones entre las palabras del texto), es de esperar que las **máquinas de vector de soporte** destaquen.

5.2. Resultados y análisis

En esta sección se analizarán primero los **resultados** de cada par de métrica y modelo, mostrando los parámetros elegidos para el modelo (cuando sea aplicable), su tasa de acierto (tanto global como específica para cada temática) y la ordenación que hace de los artículos. Tras esto, se **compararán** los resultados entre ellos mediante gráficas, para analizar el rendimiento y contrastar las hipótesis.

5.2.1. Resultados

Los resultados de cada uno de los clasificadores estudiados son los siguientes:

kNN con frecuencia absoluta

Los mejores parámetros para el clasificador estudiado son:

- **Número de vecinos**: 9
- **Tipo de pesado**: Distancia

El clasificador tiene en cuenta los 9 artículos más cercanos a la hora de clasificarlo, usando la distancia para pesar los artículos (es decir, los artículos menos parecidos tienen menos peso a la hora de la clasificación).

Las **tasas de acierto** del clasificador son las siguientes:

- **Tasa de acierto (general)**: 0.7111
  - **Deportes**: 0.6667
  - **Política**: 0.9333
  - **Salud**: 0.5333

Además, se puede observar la clasificación de los artículos realizada por el clasificador en la Tabla 5.1.

Indice	Deportes	Politica	Salud
1	Articulo 22 - Deporte (0.6715)	Articulo 25 - Politica (0.8062)	Articulo 25 - Salud (0.5705)
2	Articulo 16 - Deporte (0.6556)	Articulo 23 - Politica (0.7912)	Articulo 17 - Salud (0.5674)
3	Articulo 26 - Deporte (0.6060)	Articulo 24 - Politica (0.7125)	Articulo 29 - Salud (0.5627)
4	Articulo 19 - Deporte (0.5881)	Articulo 21 - Politica (0.7076)	Articulo 27 - Salud (0.5565)
5	Articulo 21 - Deporte (0.5574)	Articulo 18 - Politica (0.6826)	Articulo 16 - Salud (0.5144)
6	Articulo 28 - Deporte (0.5379)	Articulo 20 - Politica (0.6749)	Articulo 30 - Salud (0.4907)
7	Articulo 20 - Deporte (0.4624)	**Articulo 24 - Deporte (0.6551)**	Articulo 22 - Salud (0.4860)
8	Articulo 18 - Deporte (0.4515)	Articulo 30 - Politica (0.5882)	**Articulo 16 - Politica (0.4550)**
9	Articulo 25 - Deporte (0.4488)	Articulo 26 - Politica (0.5683)	Articulo 24 - Salud (0.4453)
10	Articulo 30 - Deporte (0.4462)	Articulo 19 - Politica (0.5505)	
11		**Articulo 20 - Salud (0.5164)**	
12		Articulo 27 - Politica (0.4988)	
13		Articulo 17 - Politica (0.4913)	
14		**Articulo 23 - Salud (0.4815)**	
15		**Articulo 23 - Deporte (0.4808)**	
16		Articulo 29 - Politica (0.4740)	
17		**Articulo 29 - Deporte (0.4711)**	
18		**Articulo 21 - Salud (0.4599)**	
19		Articulo 28 - Politica (0.4572)	
20		**Articulo 26 - Salud (0.4565)**	
21		**Articulo 27 - Deporte (0.4553)**	
22		**Articulo 28 - Salud (0.4520)**	
23		**Articulo 17 - Deporte (0.4490)**	
24		Articulo 22 - Politica (0.4480)	
25		**Articulo 19 - Salud (0.4447)**	
26		**Articulo 18 - Salud (0.4436)**	

Cuadro 5.1: Ordenación de los documentos - kNN con frecuencia absoluta

Como se puede observar claramente, los resultados del clasificador son **pobres** (al menos, comparado con el resto de clasificadores que se estudiarán).

En concreto, se puede ver que el clasificador tiene problemas a la hora de clasificar artículos de las temáticas de **Deportes** o **Salud**, teniendo un sesgo claro hacia **Política**. Esto se puede observar en las tasas de acierto (siendo la de política mucho más alta que las demás) y en los documentos clasificados (donde la gran mayoría de documentos han sido clasificados como política).

Naive Bayes con frecuencia absoluta

En este caso, no se ha ajustado ningún parámetro del clasificador.

Las **tasas de acierto** del clasificador son las siguientes:

- **Tasa de acierto (general)**: 0.8889
  - **Deportes**: 0.8667
  - **Política**: 0.8000
  - **Salud**: 1.000

Además, se puede observar la clasificación de los artículos realizada por el clasificador en la Tabla 5.2.

Indice	Deportes	Politica	Salud
1	Articulo 22 - Deporte (1.0000)	Articulo 21 - Politica (1.0000)	**Articulo 16 - Politica (1.0000)**
2	Articulo 27 - Deporte (1.0000)	Articulo 25 - Politica (1.0000)	Articulo 16 - Salud (1.0000)
3	Articulo 25 - Deporte (1.0000)	Articulo 18 - Politica (1.0000)	Articulo 17 - Salud (1.0000)
4	Articulo 20 - Deporte (1.0000)	Articulo 20 - Politica (1.0000)	Articulo 19 - Salud (1.0000)
5	Articulo 17 - Deporte (1.0000)	Articulo 23 - Politica (1.0000)	Articulo 20 - Salud (1.0000)
6	Articulo 19 - Deporte (1.0000)	Articulo 24 - Politica (1.0000)	Articulo 22 - Salud (1.0000)
7	Articulo 26 - Deporte (1.0000)	Articulo 30 - Politica (1.0000)	Articulo 24 - Salud (1.0000)
8	Articulo 21 - Deporte (1.0000)	Articulo 27 - Politica (1.0000)	Articulo 25 - Salud (1.0000)
9	Articulo 18 - Deporte (1.0000)	Articulo 26 - Politica (0.9999)	Articulo 27 - Salud (1.0000)
10	Articulo 30 - Deporte (1.0000)	Articulo 28 - Politica (0.9999)	Articulo 28 - Salud (1.0000)
11	Articulo 16 - Deporte (1.0000)	Articulo 17 - Politica (0.9999)	Articulo 29 - Salud (1.0000)
12	Articulo 23 - Deporte (1.0000)	Articulo 29 - Politica (0.9992)	Articulo 30 - Salud (1.0000)
13	Articulo 28 - Deporte (0.9855)	**Articulo 29 - Deporte (0.7650)**	Articulo 18 - Salud (1.0000)
14		**Articulo 24 - Deporte (0.5604)**	Articulo 26 - Salud (1.0000)
15			Articulo 21 - Salud (1.0000)
16			**Articulo 19 - Politica (1.0000)**
17			**Articulo 22 - Politica (1.0000)**
18			Articulo 23 - Salud (0.9987)

Cuadro 5.2: Ordenación de los documentos - Naive Bayes con frecuencia absoluta

A simple vista, estudiando la tasa de acierto se puede observar una **mejora notable** del rendimiento del clasificador, mejorando mucho el resultado en los campos de **Deportes** y **Salud**. Como se planteó en las hipótesis, este resultado era esperado al ser Naive Bayes un modelo con buenos resultados en el campo del procesamiento de lenguaje.

En este caso, el clasificador tiene un ligero sesgo hacia **Salud**, clasificando varios artículos de **Política** erróneamente. Aun así, es necesario remarcar que esos artículos (16, 19 y 22) tratan temas relacionados tangencialmente con la salud (ley de eutanasia y terapias de conversión), por lo que los artículos tienen un poco de solapamiento en sus temáticas.

Respecto a los artículos de **Deportes** clasificados erróneamente (24 y 29), estos tratan temas sobre la Copa del Rey y deportistas españoles, por lo que contienen términos que forman parte del glosario de política.

SVM con frecuencia absoluta

Los mejores parámetros para el clasificador estudiado son:

- **Kernel**: Lineal

El kernel utilizado por la máquina de vector de soporte es **lineal**, el resultado más típico cuando se utilizan estos modelos con clasificación de documentos.

Las **tasas de acierto** del clasificador son las siguientes:

- **Tasa de acierto (general)**: 0.8889
  - **Deportes**: 1.000
  - **Política**: 0.733
  - **Salud**: 0.933

Además, se puede observar la clasificación de los artículos realizada por el clasificador en la Tabla 5.3.

Indice	Deportes	Politica	Salud
1	Articulo 20 - Deporte (0.8719)	Articulo 25 - Politica (0.9632)	Articulo 20 - Salud (0.9851)
2	Articulo 17 - Deporte (0.8650)	Articulo 30 - Politica (0.9445)	Articulo 25 - Salud (0.9582)
3	Articulo 16 - Deporte (0.8572)	Articulo 23 - Politica (0.9186)	**Articulo 16 - Politica (0.9349)**
4	Articulo 26 - Deporte (0.8483)	Articulo 21 - Politica (0.8884)	Articulo 16 - Salud (0.9102)
5	Articulo 19 - Deporte (0.8363)	Articulo 24 - Politica (0.8746)	Articulo 29 - Salud (0.8951)
6	Articulo 25 - Deporte (0.8106)	Articulo 18 - Politica (0.8609)	Articulo 27 - Salud (0.8860)
7	Articulo 30 - Deporte (0.7654)	Articulo 26 - Politica (0.7234)	Articulo 17 - Salud (0.8860)
8	Articulo 22 - Deporte (0.7643)	Articulo 27 - Politica (0.6943)	Articulo 22 - Salud (0.8832)
9	Articulo 27 - Deporte (0.7572)	Articulo 20 - Politica (0.5856)	Articulo 24 - Salud (0.8464)
10	Articulo 28 - Deporte (0.7357)	Articulo 17 - Politica (0.5648)	**Articulo 22 - Politica (0.8452)**
11	Articulo 21 - Deporte (0.7303)	Articulo 29 - Politica (0.5189)	Articulo 30 - Salud (0.8359)
12	Articulo 24 - Deporte (0.6541)	**Articulo 23 - Salud (0.3685)**	Articulo 19 - Salud (0.8145)
13	Articulo 18 - Deporte (0.5890)		Articulo 28 - Salud (0.8033)
14	Articulo 23 - Deporte (0.5827)		Articulo 26 - Salud (0.7835)
15	Articulo 29 - Deporte (0.4713)		Articulo 18 - Salud (0.7643)
16	**Articulo 28 - Politica (0.4268)**		Articulo 21 - Salud (0.7278)
17			**Articulo 19 - Politica (0.6606)**

Cuadro 5.3: Ordenación de los documentos - SVM con frecuencia absoluta

Se puede ver que los resultados son **buenos** y muy parecidos a los del clasificador anterior (Naive Bayes), de nuevo siendo mejor que kNN. En este caso el rendimiento en **Política** ha empeorado considerablemente a costa de mejorar los resultados en **Deportes**, habiendo un ligero descenso también en **Salud**.

Se puede observar de nuevo que los artículos de política que tratan leyes sobre salud han sido clasificado erróneamente de nuevo. Aun así, excepto el artículo 16 (que es una entrevista), estas clasificaciones erróneas se encuentran al final de la lista, teniendo grados de pertenencia bajos.

Respecto al resto de artículos clasificados erróneamente, se puede ver que están los últimos en la lista de las temáticas, por lo que aun habiendo sido clasificados de forma errónea se puede identificar que no encajan bien en las temáticas asignadas.

kNN con TF-IDF

Los mejores parámetros para el clasificador estudiado son:

- **Número de vecinos**: 1
- **Tipo de pesado**: Uniforme

Sorprendentemente, el clasificador kNN en este caso tiene en cuenta únicamente al artículo más parecido de la base de artículos de entrenamiento, en vez de considerar varios.

Las **tasas de acierto** del clasificador son las siguientes:

- **Tasa de acierto (general)**: 0.8889
  - **Deportes**: 0.9333
  - **Política**: 0.7333
  - **Salud**: 1.000

Además, se puede observar la clasificación de los artículos realizada por el clasificador en la Tabla 5.4.

Indice	Deportes	Politica	Salud
1	Articulo 16 - Deporte (1.0000)	**Articulo 28 - Deporte (1.0000)**	**Articulo 16 - Politica (1.0000)**
2	Articulo 17 - Deporte (1.0000)	Articulo 17 - Politica (1.0000)	**Articulo 19 - Politica (1.0000)**
3	Articulo 18 - Deporte (1.0000)	Articulo 18 - Politica (1.0000)	**Articulo 22 - Politica (1.0000)**
4	Articulo 19 - Deporte (1.0000)	Articulo 20 - Politica (1.0000)	Articulo 16 - Salud (1.0000)
5	Articulo 20 - Deporte (1.0000)	Articulo 21 - Politica (1.0000)	Articulo 17 - Salud (1.0000)
6	Articulo 21 - Deporte (1.0000)	Articulo 23 - Politica (1.0000)	Articulo 18 - Salud (1.0000)
7	Articulo 22 - Deporte (1.0000)	Articulo 24 - Politica (1.0000)	Articulo 19 - Salud (1.0000)
8	Articulo 23 - Deporte (1.0000)	Articulo 25 - Politica (1.0000)	Articulo 20 - Salud (1.0000)
9	Articulo 24 - Deporte (1.0000)	Articulo 26 - Politica (1.0000)	Articulo 21 - Salud (1.0000)
10	Articulo 25 - Deporte (1.0000)	Articulo 27 - Politica (1.0000)	Articulo 22 - Salud (1.0000)
11	Articulo 26 - Deporte (1.0000)	Articulo 29 - Politica (1.0000)	Articulo 23 - Salud (1.0000)
12	Articulo 27 - Deporte (1.0000)	Articulo 30 - Politica (1.0000)	Articulo 24 - Salud (1.0000)
13	Articulo 29 - Deporte (1.0000)		Articulo 25 - Salud (1.0000)
14	Articulo 30 - Deporte (1.0000)		Articulo 26 - Salud (1.0000)
15	**Articulo 28 - Politica (1.0000)**		Articulo 27 - Salud (1.0000)
16			Articulo 28 - Salud (1.0000)
17			Articulo 29 - Salud (1.0000)
18			Articulo 30 - Salud (1.0000)

Cuadro 5.4: Ordenación de los documentos - kNN con TF-IDF

Lo primero que se observa de forma evidente es la **mejora notable** del rendimiento cuando se compara con el mismo modelo usando frecuencia absoluta. Como se había planteado en las hipótesis, el uso de TF-IDF supone un aumento notable del rendimiento del clasificador. Aun así, el rendimiento del modelo sigue siendo muy similar al de los mejores clasificadores de frecuencia absoluta.

El modelo funciona especialmente bien con artículos de **Salud**, clasificando todos correctamente. Ahora bien, el rendimiento con **Política** empeora, teniendo el problema visto hasta ahora (los artículos 16, 19 y 22 solapando entre ambas temáticas) y clasificando otro artículo (el 28) erróneamente. Respecto a **Deportes**, funciona también notablemente bien, clasificando todos los artículos (salvo el 28) de forma correcta.

Al tener en cuenta un único vecino y usar pesado uniforme, no es posible ordenar internamente los artículos, siguiendo estos un orden arbitrario (alfabéticamente).

Naive Bayes con TF-IDF

En este caso, no se ha ajustado ningún parámetro del clasificador.

Las **tasas de acierto** del clasificador son las siguientes:

- **Tasa de acierto (general)**: 0.9111
  - **Deportes**: 0.9333
  - **Política**: 0.8000
  - **Salud**: 1.000

Además, se puede observar la clasificación de los artículos realizada por el clasificador en la Tabla 5.5.

Indice	Deportes	Politica	Salud
1	Articulo 22 - Deporte (0.7718)	Articulo 25 - Politica (0.7312)	Articulo 17 - Salud (0.7962)
2	Articulo 27 - Deporte (0.7685)	Articulo 21 - Politica (0.7131)	Articulo 24 - Salud (0.7868)
3	Articulo 17 - Deporte (0.7212)	Articulo 24 - Politica (0.7106)	Articulo 29 - Salud (0.7758)
4	Articulo 19 - Deporte (0.7207)	Articulo 20 - Politica (0.6437)	Articulo 28 - Salud (0.7725)
5	Articulo 16 - Deporte (0.6890)	Articulo 27 - Politica (0.5816)	Articulo 30 - Salud (0.7579)
6	Articulo 26 - Deporte (0.6837)	Articulo 18 - Politica (0.5524)	Articulo 25 - Salud (0.7533)
7	Articulo 23 - Deporte (0.6665)	Articulo 23 - Politica (0.5456)	Articulo 16 - Salud (0.7454)
8	Articulo 21 - Deporte (0.6530)	Articulo 17 - Politica (0.5378)	Articulo 18 - Salud (0.7199)
9	Articulo 20 - Deporte (0.6176)	Articulo 30 - Politica (0.5157)	Articulo 22 - Salud (0.6938)
10	Articulo 28 - Deporte (0.5454)	Articulo 29 - Politica (0.4554)	Articulo 26 - Salud (0.6532)
11	Articulo 30 - Deporte (0.5319)	Articulo 26 - Politica (0.4340)	Articulo 21 - Salud (0.6530)
12	Articulo 25 - Deporte (0.5007)	Articulo 28 - Politica (0.4332)	**Articulo 16 - Politica (0.6504)**
13	Articulo 18 - Deporte (0.4956)	**Articulo 24 - Deporte (0.3958)**	Articulo 20 - Salud (0.6304)
14	Articulo 29 - Deporte (0.3632)		Articulo 19 - Salud (0.6117)
15			Articulo 27 - Salud (0.5957)
16			Articulo 23 - Salud (0.5555)
17			**Articulo 22 - Politica (0.5263)**
18			**Articulo 19 - Politica (0.5071)**

Cuadro 5.5: Ordenacion de los documentos - Naive Bayes con TF-IDF

Se puede ver un **aumento ligero** en el rendimiento del clasificador comparado con todos los demás, obteniendo una tasa de acierto muy elevada (de alrededor del 91 %). Esto, de nuevo, se ajusta a lo esperado (Naive Bayes es un buen modelo para clasificación documental y TF-IDF es una mejor métrica).

La temática con mejor clasificación es, de nuevo, **Salud**, clasificando correctamente todos sus artículos y teniendo un ligero sesgo (los artículos de **Política** que tratan temas de salud acaban siendo clasificados también). El único error que se encuentra en **Deporte** es el artículo 24 (tratando la Copa del Rey y conteniendo palabras del glosario de Política), clasificado como Política. Por lo demás, todos los artículos se clasifican adecuadamente.

Se puede observar además que, en el orden de los documentos, los artículos clasificados erróneamente se encuentran hacia el final de la lista, demostrando que el clasificador es capaz de detectar que no encajan totalmente en la temática.

SVM con TF-IDF

Los mejores parámetros para el clasificador estudiado son:

- **Kernel**: Lineal

Igual que en el caso con frecuencia absoluta, el kernel utilizado es lineal, lo más típico en casos de procesamiento de lenguaje.

Las **tasas de acierto** del clasificador son las siguientes:

- **Tasa de acierto (general)**: 0.9333
  - **Deportes**: 1.000
  - **Política**: 0.800
  - **Salud**: 1.000

Además, se puede observar la clasificación de los artículos realizada por el clasificador en la Tabla 5.6.

Indice	Deportes	Politica	Salud
1	Articulo 17 - Deporte (0.9877)	Articulo 25 - Politica (0.9790)	Articulo 17 - Salud (0.9821)
2	Articulo 16 - Deporte (0.9863)	Articulo 24 - Politica (0.9684)	Articulo 24 - Salud (0.9752)
3	Articulo 19 - Deporte (0.9768)	Articulo 21 - Politica (0.9359)	Articulo 30 - Salud (0.9713)
4	Articulo 22 - Deporte (0.9745)	Articulo 30 - Politica (0.9348)	Articulo 18 - Salud (0.9675)
5	Articulo 27 - Deporte (0.9580)	Articulo 27 - Politica (0.9259)	Articulo 16 - Salud (0.9670)
6	Articulo 30 - Deporte (0.9513)	Articulo 20 - Politica (0.9088)	Articulo 28 - Salud (0.9649)
7	Articulo 26 - Deporte (0.9508)	Articulo 23 - Politica (0.8948)	Articulo 22 - Salud (0.9612)
8	Articulo 21 - Deporte (0.8929)	Articulo 29 - Politica (0.8911)	Articulo 25 - Salud (0.9571)
9	Articulo 20 - Deporte (0.8793)	Articulo 26 - Politica (0.8801)	Articulo 29 - Salud (0.9563)
10	Articulo 23 - Deporte (0.8114)	Articulo 17 - Politica (0.8792)	Articulo 20 - Salud (0.9475)
11	Articulo 28 - Deporte (0.7647)	Articulo 18 - Politica (0.8604)	Articulo 26 - Salud (0.9405)
12	Articulo 25 - Deporte (0.7356)	Articulo 28 - Politica (0.6981)	Articulo 21 - Salud (0.9262)
13	Articulo 29 - Deporte (0.6724)		Articulo 19 - Salud (0.8836)
14	Articulo 18 - Deporte (0.6680)		**Articulo 16 - Politica (0.8475)**
15	Articulo 24 - Deporte (0.6084)		**Articulo 19 - Politica (0.8441)**
16			Articulo 23 - Salud (0.8369)
17			Articulo 27 - Salud (0.8099)
18			**Articulo 22 - Politica (0.7725)**

Cuadro 5.6: Ordenacion de los documentos - SVM con TF-IDF

El rendimiento del clasificador ha **aumentado** de nuevo, siendo la tasa de acierto muy elevada (alrededor del 93 %) y siendo éste el **mejor clasificador** estudiado. Esto se ajusta totalmente a las hipótesis planteadas (SVM siendo el mejor modelo y TF-IDF la mejor métrica).

El rendimiento en **Deportes** y **Salud** es muy bueno, siendo capaz de clasificar adecuadamente todos los artículos de esas temáticas. Los únicos artículos clasificados de forma errónea son, de nuevo, los de **Política** (16, 19 y 22), al solapar sus temáticas entre Política y Salud. Estos errores

han sido consistentes en prácticamente todos los clasificadores estudiados, y se pueden achacar al contenido de los propios artículos (siendo estos ambiguos).

Finalmente, se puede observar en la ordenación que las clasificaciones erróneas se encuentran al fondo de la lista. Una vez más, el clasificador es capaz de identificar artículos con un grado de pertenencia bajo a la temática.

5.2.2. Comparativa

De cara a comparar los clasificadores entre sí, se estudiarán sus **tasas de acierto** comparándolas directamente. Para mayor simplicidad, se puede observar una recopilación de la tasa de acierto de todos los clasificadores en la Tabla 5.7.

Modelo		**General**	**Deportes**	**Politica**	**Salud**
Frecuencia absoluta	**kNN**	0.71	0.67	0.93	0.53
	Naive Bayes	0.89	0.87	0.8	1.00
	SVM	0.89	1.0	0.73	0.93
TF-IDF	**kNN**	0.89	0.93	0.73	1.00
	Naive Bayes	0.91	0.93	0.8	1.00
	SVM	**0.93**	**1.00**	**0.8**	**1.00**

Cuadro 5.7: Tasa de acierto de los modelos

Como se puede observar en la tabla, **TF-IDF mejora notablemente el resultado** cuando se compara con la métrica de frecuencia absoluta, obteniendo tasas de acierto en general más elevada. Además, **Naive Bayes y especialmente las máquinas de vector de soporte ofrecen los mejores resultados**, siendo el mejor clasificador la **máquina de vector de soporte usando TF-IDF**.

Para estudiar mejor las tasas de acierto, se usarán una serie de gráficas comparando los resultados de cada clasificador visualmente.

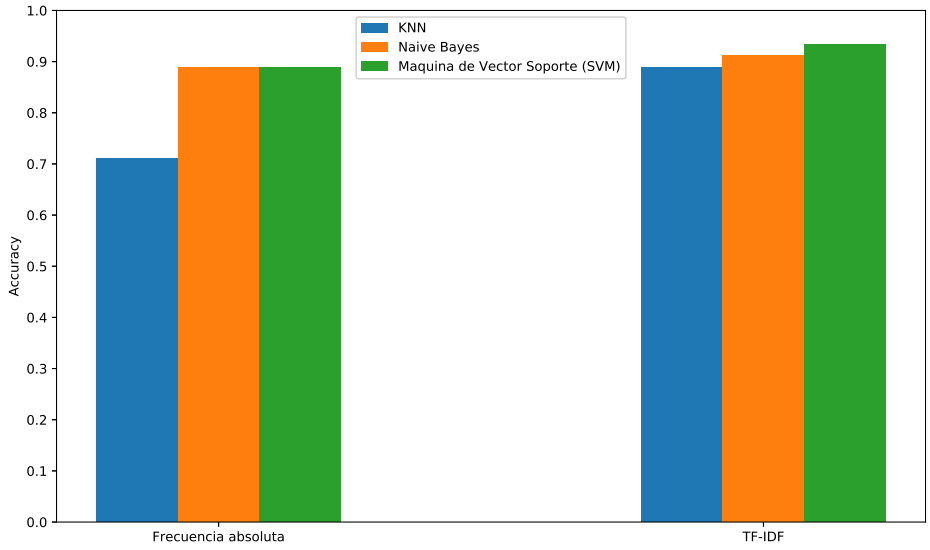

Figura 5.1: Tasa de acierto - General

En la Figura 5.1 se observa la **tasa de aciertos general** de todos los clasificadores. Como se puede ver, salvo kNN usando frecuencia absoluta que tiene un rendimiento ligeramente peor, **todos los clasificadores obtienen buenos rendimientos**. Además, se ve claramente la mejora al usar **TF-IDF**, siendo todas las tasas de acierto superiores a las de frecuencia absoluta y siendo el mejor clasificador una máquina de vector de soporte con TF-IDF. También se puede observar que, en general, **kNN ofrece resultados peores** cuando se comparan con el resto de modelos, siendo Naive Bayes y especialmente las máquinas de vector de soporte superiores.

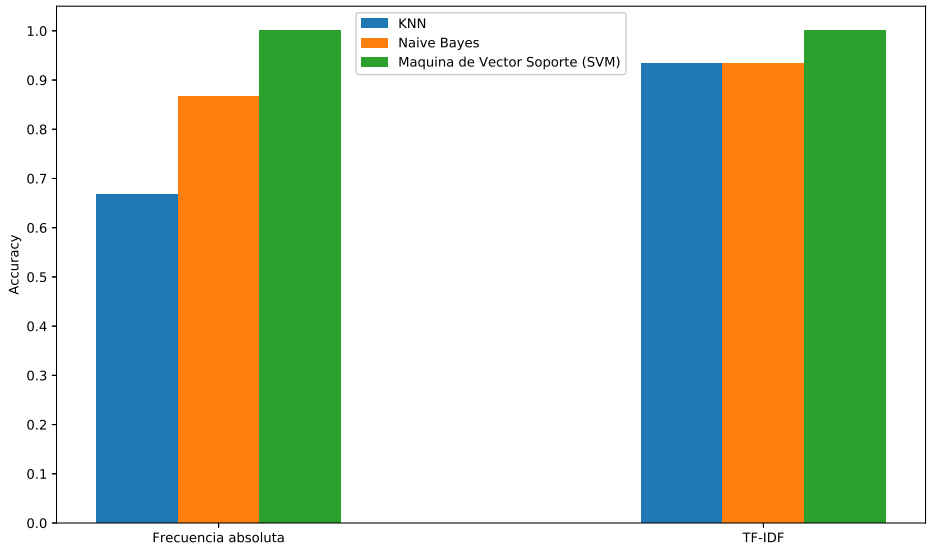

Figura 5.2: Tasa de acierto - Deportes

Estudiando el acierto a la hora de clasificar artículos de **Deportes** (como se puede ver en la Figura 5.2), se puede ver que **las máquinas de vector de soporte ofrecen el mejor rendimiento para este campo** independientemente de la métrica utilizada, clasificando correctamente

todos los artículos de esta temática. Como contraste, **los modelos que usan frecuencia absoluta clasifican peor los artículos de deportes**, siendo el rendimiento de kNN especialmente bajo (apenas llegando al 70 %).

Comentando el acierto a la hora de clasificar artículos de **Política** (como se observa en la Figura 5.3), se observa algo inesperado: **comparado con el resto de temáticas, los artículos de política son mal clasificados**, rondando todas las tasas de acierto alrededor del 80 %. La única excepción es **kNN con frecuencia absoluta** que llega a superar el 90 %, pero como ya se comentó a la hora de analizar los resultados esto se debe a un sobreajuste claro del clasificador, clasificando la gran mayoría de artículos como política.

La razón para estos artículos, como se comentó durante el análisis individual de los clasificadores, son **los artículos 16, 19 y 22**. Estos artículos tratan temas que se solapan entre política y salud (concretamente la ley de la eutanasia y gente defendiendo las terapias de conversión como tratamiento médico). Por tanto, el problema de estos clasificadores podría ser resuelto posiblemente con un glosario más extenso, que permita distinguir de forma más exacta los temas.

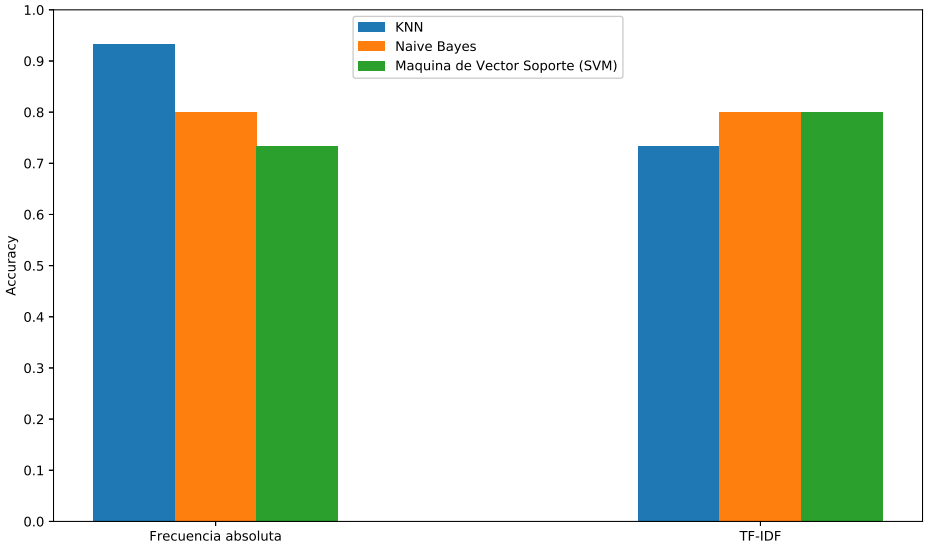

Figura 5.3: Tasa de acierto - Política

Finalmente, estudiando el acierto a la hora de clasificar artículos de **Salud** (como se observa en la Figura 5.4), lo más evidente es que **TF-IDF destaca clasificando artículos de esta temática**, clasificando sin fallos todos los artículos de salud independientemente del modelo. En cambio, **kNN con frecuencia absoluta tiene problemas notables para clasificar artículos de salud**, apenas superando el 50 % de tasa de acierto. El resto de modelos usando frecuencia absoluta (Naive Bayes y máquina de vector de soporte) ofrecen resultados buenos, superiores al 90 %.

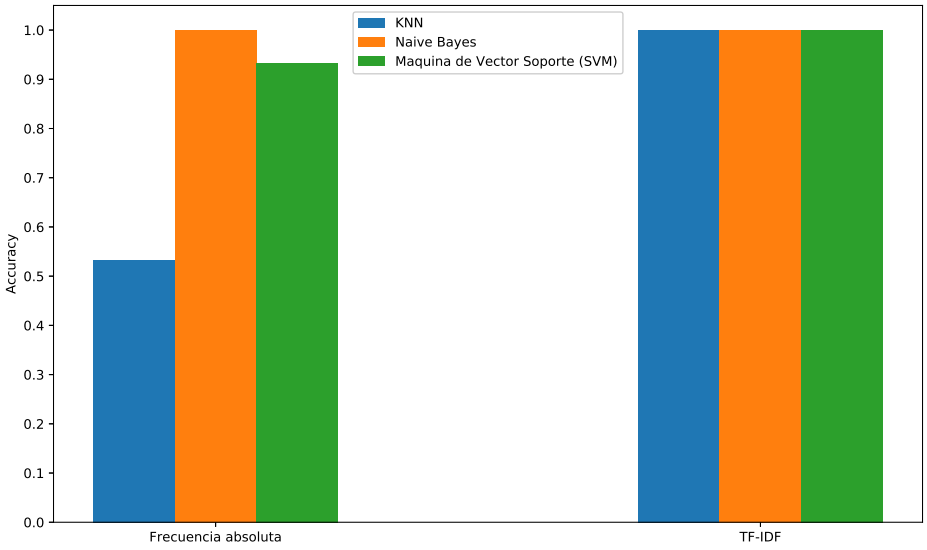

Figura 5.4: Tasa de acierto - Salud

5.3. Validación de hipótesis

Tras analizar todos los resultados obtenidos, se pueden extraer las siguientes conclusiones respecto a las hipótesis:

- **Los glosarios obtenidos son representativos de cada temática:**

  Esta hipótesis es la más **dudosa**. Los resultados obtenidos son, en general, muy buenos (teniendo el mejor clasificador una tasa de acierto de aproximadamente 93 %). Ahora bien, todos los fallos se deben a artículos cuya temática se encuentra solapada entre dos de las temáticas propuestas.

  Posiblemente usando **un glosario más extenso y específico** (con separaciones más concretas entre temáticas) esos artículos se hubieran podido clasificar de forma adecuada, obteniendo mejores tasas de acierto. Otra posibilidad es simplemente asumir que el clasificador encontrará artículos con temática ambigua y, por tanto, es un error aceptable y asumible si se quiere evitar un sobreajuste excesivo.

- **La métrica TF-IDF dará mejores resultados que la frecuencia absoluta:**

  Esta hipótesis se puede considerar **validada**. Observando la tasa de acierto general de cada modelo, se puede ver claramente que **el peor modelo de TF-IDF sigue teniendo una tasa de acierto igual que la mejor tasa de acierto usando frecuencia absoluta**, teniendo además varios modelos que mejoran su rendimiento.

- **Los modelos de Naive Bayes y de máquinas de vector de soporte darán mejores resultados que kNN:**

De nuevo, esta hipótesis se puede considerar **validada**. Para ambas métricas, **el modelo con peor rendimiento es kNN** (siendo el rendimiento notablemente peor usando frecuencia absoluta y ligeramente mejor usando TF-IDF). En cambio, tanto Naive Bayes como las máquinas de vector de soporte ofrecen los mejores resultados para ambas métricas.

- **El modelo que mejores resultados ofrecerá es la máquina de vector de soporte:**

  Esta hipótesis se puede **validar**. El mejor clasificador estudiado ha sido la **máquina de vector de soporte usando TF-IDF**, teniendo la mayor tasa de acierto comparada con todos los demás clasificadores. En el caso de la frecuencia absoluta, el mejor resultado también lo ofrece la máquina de vector de soporte (si bien su rendimiento está empatado con el de Naive Bayes).

Capítulo 6

Conclusiones

En este trabajo se ha descrito el desarrollo de un **clasificador documental**, detallando los principales pasos: la **selección de documentos a realizar**, la **extracción realizada del glosario** y el **desarrollo del clasificador** como tal. Además, se ha realizado una **experimentación** con susodicho clasificador, estudiando varias métricas y modelos con el fin de identificar la combinación que ofrece el mejor rendimiento.

Algunas conclusiones que se pueden extraer de este trabajo son:

- Se ha hecho una **elección de artículos** variada y adecuada, funcionado los clasificadores propuestos de forma adecuada usando esos documentos.
- La **elección del glosario** realizada en el trabajo ha sido adecuada, obteniendo glosarios **extensos** y (salvo excepciones) **capaces de distinguir adecuadamente cada temática** sin problemas de solapamientos.
- Los **clasificadores propuestos** ofrecen resultados, en general, **muy buenos** para este problema concreto. Específicamente, el mejor clasificador (**máquina de vector de soporte con TF-IDF**) ofrece una tasa de acierto muy elevada del 93 %, siendo además los únicos errores en artículos cuya temática resulta demasiado ambigua.

Algunas posibles lineas futuras para este trabajo serían:

- Utilizar **un glosario más extenso y refinado**. Los problemas de clasificación se deben a problemas de ambigüedad, por lo que sería interesante estudiar si podrían ser solventados mediante un glosario más refinado que distinga de forma más concreta cada temática.
- Probar el rendimiento del clasificador con **un mayor número de temáticas y documentos**, menos distinguibles entre ellos. En problemas reales no es típico que el número de clases sea tan bajo y tan fácilmente distinguible, por lo que resultaría de interés estudiar el rendimiento de estos modelos frente a una mayor variedad de temáticas.
- Utilizar **modelos más avanzados** como redes neuronales (tanto *shallow* como *deep*). Estos modelos de clasificación son el estado del arte actualmente para muchos campos del procesamiento de lenguaje natural, siendo algunos ejemplos de modelos usados **GPT-3** o **BERT**.

Apéndice A

Contenidos del fichero entregable

En el fichero comprimido entregado, se encuentran disponibles los siguientes elementos:

- Memoria del trabajo en formato **PDF**, con nombre *Ingenieria Linguistica - Practica 2.pdf*.
- Código fuente del **extractor de glosarios** en **Python** en el fichero *extractor_terminologico.py*.
- Código fuente del **clasificador de documentos** en **Python** en el fichero *clasificador_documental.py*.
- Lista de *stopwords* utilizadas por el extractor de glosarios y el clasificador de documentos en formato **TXT** en el fichero *stopwords.txt*.
- Todos los artículos seleccionados en formato **TXT** en el directorio *"Articulos"*.
- Artículos seleccionados para la extracción de glosarios en el directorio *"Glosario"*.
- Artículos seleccionados para ser clasificados en el directorio *"Por_clasificar"*.
- Gráficas utilizadas en la memoria en mayor resolución en el directorio *"Gráficas"*.