Bitácora Preliminar

Materia: Análisis y Procesamiento Inteligente de Texto

Maestro: Octavio Augusto Sánchez Velázquez

Alumno: José Antonio Velázquez Sánchez

Entradas de la Bitácora

Introducción:

Esta es la bitácora que fui llevando al inicio del proyecto antes de comenzar a trabajar directamente en los cuadernos de Jupyter.

En general aquí detallo el proceso de obtener el corpus. Mucho de lo que se dice aquí es resumido en el primer cuaderno de jupyter.

12/Nov: Cambio de proyecto

- El día de hoy comenté con el maestro la posibilidad de cambiar de proyecto.
- Tenía pensado hacer algo que involucrara traducción automática y corpus multilingües, pero debido a a cuestions de tiempo y falta de recursos preferí cambiarlo.
- Ahora mi nuevo proyecto se trata de minar tópicos desde archivos de subtítulos para hacer clasificadores de películas y además quizás encontrar alguna manera de asociar los vocabularios en las distintas lenguas que usaré.

14/Nov: Exploracion de herramientas para descargar subtítulos.

- Tras ver varias opciones decidí utilizar esta opción que es un wrapper de la API de opensubtitles escrito en python.

16/Nov: Sobre filtrar películas por su género

Para obtener mi corpus de subtítulos es primero necesario decidir de cuáles películas quiero obtener los subtítulos. La idea que tengo es filtrar las películas listadas en IMDB por género y ordenarlas por número de votantes (por ejemplo, como se hace aquí: https://www.imdb.com/search/title?genres=comedy&sort=num_votes,desc&explore=title_type,genres&view=simple) .

Quiero hacer ese query en específico ya que necesitaré una lista de películas separadas por género, y como también tiene que haber subtítulos disponibles en las tres lenguas que utilizo, necesito que las películas sean lo suficientemente populares para que existan esos subtítulos dentro de opensubtitles.org.

Aunque IMDb tiene una métrica de “Popularidad”, lo que no me termina de gustar es que ésta métrica

se refiere a una popularidad que cambia con el tiempo y favorece a las películas más recientes, e incluso a películas que no se han estrenado. Luego, sospecho, las películas más populares quizás no tengan subtítulos disponibles en opensubtitles.org en las tres lenguas. Es por esto que prefiero la métrica de “número de votos” que ha tenido la película para considerar su nivel de “popularidad”.

16/Nov: Sobre encontrar un API para hacer la query de películas

Hasta el momento no he encontrado una API de IMDb (o de sitios similares) que permita hacer el query que he determinado (filtrar por género y ordenar por número de votantes).

Por un lado está imdbpy que permite hacer búsqueda de películas con el título o con keywords, pero no permite hacerla por género y tampoco parece permitir ordenar las búsquedas por votantes.

Por otro lado está themoviedb que parece permitir filtrar por género, pero no ordenar por número de votantes (aunque sí por el índice de popularidad de IMDb).

De pronto la solución más sencilla que se me ocurre (aunque un poco revoltosa) es directamente extraer los IDs de las películas que aparecen al hacer una búsqueda como esta en IMDB.

16/Nov: Sobre qué géneros utilizar

Para mantener las cosas simples, creo que lo mejor que puedo hacer es tomar un número pequeño de géneros y que estén bien diferenciados entre ellos. De momento he decidido tomar 3 géneros: Romance-Comedia, Acción y Horror.

Esta decisión surge tras explorar manualmente la sección de géneros de IMDb. Mi método fue entrar a cada género, ordenar por número de votantes, y examinar rápidamente la lista de las primeras 50 películas que aparecían.

En principio, los géneros que contempla IMDb son estos:

Pero adicionalmente ofrece "géneros combinados" como acción-comedia y romance-comedia. Las películas pueden tener 1 o más géneros.

Según noté, existe un gran sobrelape entre distintos pares de géneros. Por ejemplo, prácticamente todas las películas del género “Superhero” también pertenecen al género de acción, al de aventura y al de fantasía. Buena parte de las películas del género de fantasía también están clasificadas en el género “family” y “animation”. Etc...

Debido a esto, creo conveniente escoger géneros que estén bien separados, porque presiento que esto ayudará a que la minería de tópicos tenga resultados más notables y que la clasificación también se pueda hacer de manera más acertada.

Por eso escogí Romance-Comedia, Acción y Horror, ya que considero que no hay tanto sobrelape entre estos géneros.

Entonces concluyo que haré scrapping de estas páginas:
Búsqueda en IMDb para películas de comedia-romance ordenadas por número de votantes
Búsqueda en IMDB para películas de acción, ordenadas por número de votantes
Búsqueda en IMDB para películas de horror, ordenadas por número de votantes

16/Nov - Sobre determinar lista de películas a utilizar

Tras decidir que haré un scrapping directamente de la página web de IMDb, me dediqué a hacer el código de scrapping que se puede encontrar en la carpeta de ./download-scripts.

Para ello utilicé BeautifulSoup. Hacía mucho que no lo usaba, así que tuve que dedicarme un rato a recordar cómo usarlo. Resultó sencillo obtener la información de las películas enlistadas en las

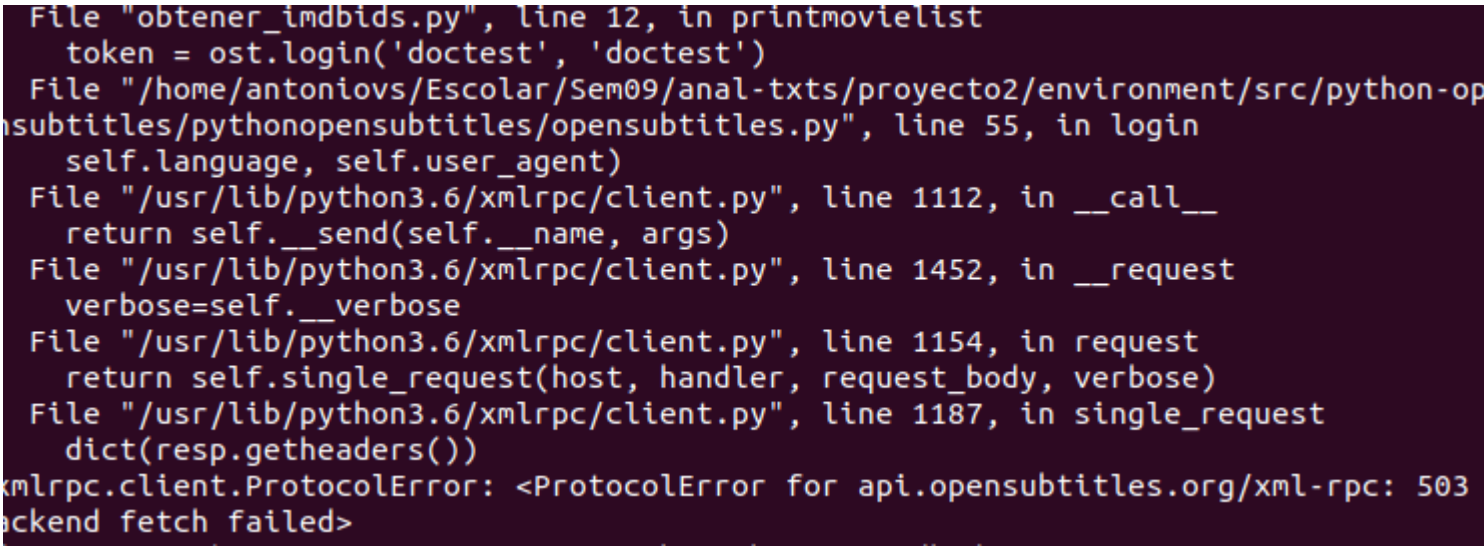
búsquedas por género. Recordé que solamente necesitaba obtener el ID de IMDb para luego hacer la descarga de subtítulos, pero decidí también extraer el nombre de la película para tener a la mano los títulos de las películas que estaría analizando.

Decidí extraer 300 títulos por cada género. Aunque aún no sé si trabajaré con todos esos subtítulos, creo que es mejor extraerlos todos desde ahora y luego ya decidir con cuántos trabajaré. Un pequeño problema al respecto del número de títulos, fue que las búsquedas de IMDb regresan 50 resultados, y no encontré cómo hacer que aparecieran más. Sin embargo se solucionó rápido al incluir un parámetro "?start=51" en la petición GET para obtener los resultados del 51 al 100... Luego solo era cuestión de iterar hasta obtener 300 títulos.

Un problema que encontré fue que algunos de los resultados enlistados en IMDb eran series de TV y no películas. No encontré una manera rápida de solucionar esto directamente en el GET request, así que me fue necesario instalar imdbpy para checar si cada resultado obtenido era una película o una serie. Decidí solo quedarme con las películas para evitar futuras complicaciones al obtener subtítulos.

Luego también pensé que quizás sería necesario revisar si en opensubtitles existían subtítulos en todas las lenguas que contemplo para las películas enlistadas. Así que añadí al script que por cada película enlistada se revisara si existían los subtítulos en opensubtitles.

Entonces, corrí el script un par de veces haciendo una que otra modificación, pero después de un par de horas comencé a tener este mensaje de error:



Por lo que entiendo, la API de opensubtitles me mandaba un error HTTP 503, el cual significa que el servidor de opensubtitles no podía atender mis peticiones ya que, quizás, se encontraba sobrecargado o en mantenimiento. Así que modifiqué el script para que no hiciera esta verificación de si los títulos obtenidos contaban con subtítulos en opensubtitle.org. Aunque esto lo pude solucionar rápido, me

preocupa que más adelante vuelva a tener errores 503 cuando comience a descargar los subtítulos y ello me cause retrasos en mi trabajo.

Así que volví a correr el script para los 3 géneros, ignorando la verificación de existencia de subtítulos, y más o menos como en 30 minutos ya obtuve las 3 listas de películas que me interesan.

16/Nov Sobre descargar las películas seleccionadas

Una vez que ya obtuve las listas de películas que consideraré para mi proyecto, me dispuse a hacer un script para descargarlas. Me basé en el script que hice el otro día cuando exploraba cómo utilizar la api de OpenSubtitles usando el modulo de python-opensubtitles. La idea es simplemente abrir cada lista, recorrer linea por linea, y descargar los subtitulos en las 3 lenguas de mi interés. Decidí que la mejor manera de organizar los subtitulos era siguiendo la estructura "./dataset/<c>/<movieid>/<subid>.<lang>.srt" donde "c" es la categoría de la película (0 para acción, 1 para romance-comedia y 2 para horror), "movieid" es el id de la película en IMDb, "subid" es el ID del subtítulo en OpenSubtitles, y "lang" es el lenguaje del subtítulo (eng, spa o fre).

Decidí además que mi script debería permitir seleccionar un rango de películas para las cuales descargar subtítulos. Esto para permitirme el descargarlos de manera progresiva, y poder reanudar descargas en caso de que vuelva a suceder el error 503. De momento decidí descargar solo los primeros 10 de cada lista para comenzar a hacer pruebas. Afortunadamente ya me apareció el mensaje 503 y se descargaron los subtítulos de las 30 películas en menos de 10 minutos.

Mi plan es ir viendo cómo preprocesaré los subtítulos mientras dejo descargando los primeros 50 subtítulos de las listas (sin tener que volver a descargar los primeros 10 que ya hice). Y luego dejar descargando los que falten para completar los 100 primeros. Después intentaré tener los 300 descargados por si las dudas.

Sin embargo, creo que la API de opensubtitles disponía un máximo de 200 descargas de subtítulos por día, por ello creo que debo de moderar mi uso de la API.

P.D. Tras descargar los subtítulos de las primeras 20 películas de cada lista, llegué al límite de 200 descargas. Debido a esto, y a los días que me quedan para entregar el proyecto, creo que lo mejor que puedo hacer es descargar los primeros 100 de cada lista en las 3 lenguas, y luego descargar todos los 300 de las 3 listas pero solo en inglés.

16/Nov Sobre cómo preprocesar y utilizar los subtítulos descargados

Me encuentro algo confundido sobre cómo utilizar los subtítulos con los que trabajaré.

Para preprocesarlos me resulta claro que debo quitar los timestamps propios de los archivos de subtítulos. Además de lematizar y quitar stopwords. Creo que para esto último usaré las herramientas de NLTK para las tres lenguas. También creo conveniente quitar todo signo de puntuación y todo número.

Pero lo que no sé es si debería de usar los archivos de subtítulos completos a la hora de obtener los tópicos, o si debería segmentar los archivos de subtítulos en partes regulares y luego obtener los tópicos tomando cada segmento como un documento.

La idea me surge porque tengo el presentimiento de que la obtención de tópicos funciona mejor con documentos relativamente cortos, y no con documentos tan grandes como todos los diálogos de una película. Pero quizás me equivoco.

Otra razón que tengo para querer segmentar los subtítulos es para posteriormente entrenar el clasificador de películas, ya que siento que si tengo más datos podrá ser mejor la clasificación.

Y finalmente, también creo que sería buena idea segmentar los subtítulos para así luego hacer mezclas entre subtítulos de las distintas lenguas, y que quizás se mantenga un poco más de relación entre las palabras que sean traducciones unas de otras.

Por otro lado, no sé si tener más documentos aumentaría el tiempo de entrenamiento... o no.

Por el momento creo que lo mejor que puedo hacer es tratar a cada subtítulo como un documento completo y luego ya ver si será necesario segmentarlos.

Así que ahora me dispongo a hacer un script que preprocese los archivos de subtitulos, y tenga como salida, por cada archivo de subtitulo, otro archivo de texto que contenga solo las palabras que pasen el filtrado.

16/Nov Sobre los lematizadores

Para mi sorpresa, parece que NLTK solo tiene implementado un lematizador para inglés, por lo que me será necesario encontrar otra herramienta para lematizar en español y francés.

Al parecer spacy tiene lematizador en diversas lenguas pero todos están basados en lexicones y solo el lematizador en inglés utiliza reglas + lexicón. Debido a esto quizás de igual si utilizo lexicones directamente para lematizar.

Encontré recomendado este conjunto de lexicones en github, que contiene pares lema-token en diversas lenguas. Quizás pueda utilizarlo.

Otra herramienta que he encontrado recomendada es el Lematizador LEFF para francés que es una herramienta open source que es un lematizador que combina el "Lexique des Formes Fléchies du Français" con reglas. Podría usarlo, en conjunto de nltk o spacy para inglés, y sería cuestión de encontrar otro lematizador para español.

También encontré recomendado a Freeling. Es una herramienta de la que he escuchado mucho, pero nunca la he usado. Recuerdo que alguna vez intenté instalarla, pero tuve problemas con eso, y creo que al final no necesirté usarlo. Sin embargo, parece que sí tiene lematizadores para las 3 lenguas y por eso quizás sea una buena opción.

Y otra opción sería que en lugar de lematizar utilice stemming. Me parece que nltk tiene stemming para las 3 lenguas con las que trabajo, pero que no es tan bueno.

Por cuestiones de tiempo creo que mejor me iré por usar lexicones para hacer la lematización. Me parece que es el camino más rápido, y de pronto mi prioridad es pasar a la obtención de tópicos y clasificación de películas. Si me da tiempo, creo que también consideraría usar Freeling.

17/Nov Sobre lematizar y sobre stopwords

- Tras decidir que lematizaría usando lexicones de pares palabra-lema, me puse a ver qué lexicón usar.
- Al final decidí usar el que ya había puesto que encontré en github. Auque también encontré el lexicón que parece ser usado por Freeling, pero solo para español e inglés
- Sin embargo me llama la atención que en los lexicones que encontré el número de pares lema-palabra en inglés son mucho menores que en español. Supongo que esto es porque en inglés no hay tantas maneras distintas de conjugar los verbos, y además los adjetivos no varían según el género.
- Sobre los stopwords, también encontré varias listas de palabras para cada una de las lenguas con las que trabajaré, pero he decidido tomar estas listas que encontré en github.
- Entre tanto, no termino de fiarme de la manera en la que lematizaré las palabras, por lo que me quedará como pendiente en el futuro revisar Freeling.

17/Nov Primera prueba lematizando

- Hice la primer prueba utilizando las listas de pares palabra-lema y las listas de stopwords que ya indiqué.
- Lo primero que noto es el tamaño pequeño del archivo limpio, que resulta de limpiar un archivo de subtítulos. Pensaba que todos los diálogos condensados serían un documento grande, pero resulta mucho más pequeño de lo que esperaba. Por alguna razón creo que esperaba que el resultado de cada archivo de subtítulos sería de la extensión de una novela corta, o algo así, pero ahora me doy cuenta que los diálogos, por si solos, no suelen ser muy extensos (en especial porque no ofrecen ninguna descripción de los escenarios o los personajes).
- Antes, cuando pensaba que la extensión sería mucho más larga, pensaba que sería necesario fragmentar los archivos en documentos más pequeños para hacer la minería de tópicos y las mezclas multilingues. Pero ahora no estoy seguro de que eso sea necesario.
- También me doy cuenta de que la lematización con las listas palabra-lema no es muy buena.
- También me doy cuenta de que hay algo de ruido en la salida. Decidí que para limpiar los textos debía capturar solo cadenas de caracteres alfabéticos, respetando los acentos de francés y español, pero eliminando signos de puntuación y apóstrofes. Esto hace que en inglés cadenas como “we'll” sea separada en “we ll” o que en francés “j'aime” se separe en “j aime”. Y luego que en la salida aparezcan cosas como “ll” “j” “re” “m” de manera muy repetida. También elimino todos los números, entonces nombres como “R2D2" (de la saga de starwars) se separe en “r d". Y finalmente, algunos tags propios de los subtítulos como “<i>” o “<b>" son filtrados como “i” o “b” respectivamente.
- Debido a lo anterior decidí eliminar cualquier palabra que, tras ser filtrada, tuviera un tamaño igual a 1. Creo que es adecuado, ya que normalmente las “letras sueltas” serían preposiciones, contracciones de pronombres o verbos, o elementos de tags, que no aportarían mucho a mi aplicación. Además, decidí añadir a los stopwords esas partículas como “ll” o “re” que aparecen de contracciones verbales (como will y are) que considero que deberían estar en las stopwords.
- También estoy considerando añadir otras palabras a las stopwords, pero no estoy seguro. Por ejemplo “www” suele aparecer en varios archivos, al final, ya que los autores de los subtítulos suelen poner la URL a su sitio web. También hay muchas palabras que siento que se repiten mucho y no aportan mucho, como “aún” u “oh”. Pero creo que me será mejor correr un script para revisar las frecuencias de las palabras y decidir si añado más palabras a las stopwords.

17/Nov Estadísticas de la primer prueba

Nota: Estas estadísticas están basadas solo en los subtítulos de las primeras 20 películas de cada género. En unos días que tenga descargadas las primeras 100 volveré a obtener estas estadísticas y las graficaré en un jupyter notebook para la entrega final.

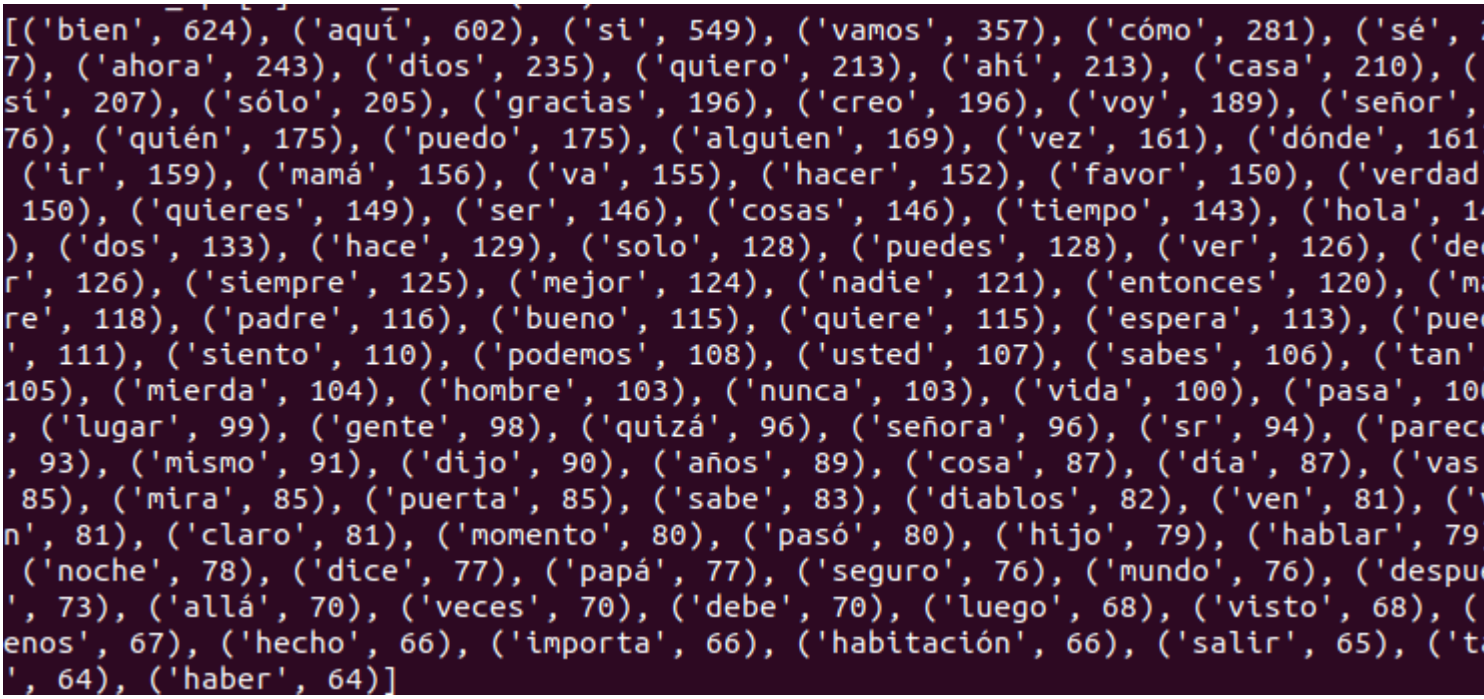
- Las 100 palabras más frecuentes en español son:
→ En el género de acción

```
[('si', 789), ('aquí', 592), ('bien', 506), ('ahora', 399), ('vamos', 360), ('sólo'
 300), ('así', 290), ('sé', 284), ('cómo', 273), ('ser', 271), ('señor', 268), ('ha
er', 262), ('puede', 257), ('dónde', 246), ('vez', 245), ('puedo', 239), ('quiero',
234), ('quién', 226), ('sr', 222), ('verdad', 214), ('tiempo', 214), ('solo', 210),
('va', 207), ('hombre', 206), ('padre', 205), ('bueno', 204), ('tan', 197), ('voy',
191), ('vida', 182), ('ahí', 178), ('nunca', 176), ('mundo', 171), ('creo', 169), (
mejor', 168), ('sabes', 167), ('gracias', 167), ('ver', 166), ('hace', 166), ('pued
s', 162), ('entonces', 160), ('siento', 155), ('dos', 152), ('alguien', 151), ('año
', 147), ('favor', 145), ('hecho', 142), ('nadie', 138), ('gente', 134), ('necesito
, 131), ('día', 128), ('siempre', 128), ('wayne', 126), ('cosas', 121), ('ciudad',
20), ('quieres', 119), ('vas', 119), ('dios', 119), ('menos', 116), ('allí', 115),
'mal', 113), ('parece', 112), ('quiere', 110), ('hombres', 109), ('amigo', 109), ('
asa', 108), ('crees', 107), ('tal', 107), ('muerte', 106), ('hijo', 102), ('usted',
102), ('momento', 100), ('sabe', 100), ('mismo', 100), ('tres', 99), ('toda', 99),
'gran', 98), ('ud', 97), ('lugar', 95), ('visto', 94), ('nave', 94), ('noche', 93),
('cada', 92), ('espera', 92), ('podemos', 91), ('luke', 91), ('mierda', 91), ('ráp
', 90), ('pasa', 90), ('dijo', 88), ('claro', 88), ('trabajo', 87), ('ir', 86), ('
oder', 86), ('cosa', 85), ('podría', 85), ('mira', 85), ('dice', 84), ('aún', 84),
'sueño', 84), ('debe', 83)]
```

→ En el género de romance

```
[('bien', 1227), ('si', 776), ('aquí', 662), ('sé', 599), ('quiero', 494), ('gracia
', 491), ('así', 490), ('bueno', 475), ('cómo', 471), ('hola', 462), ('sólo', 454),
('vamos', 449), ('ahora', 404), ('creo', 396), ('puedo', 387), ('voy', 373), ('dios
, 361), ('sabes', 344), ('siento', 327), ('ser', 324), ('quieres', 320), ('vez', 30
), ('hacer', 307), ('va', 287), ('verdad', 283), ('ir', 263), ('mejor', 262), ('qui
n', 254), ('vida', 254), ('favor', 251), ('tan', 250), ('nunca', 240), ('dos', 231)
 ('hace', 229), ('años', 224), ('tiempo', 216), ('día', 214), ('acuerdo', 214), ('c
sa', 213), ('gusta', 213), ('siempre', 212), ('ver', 210), ('alguien', 205), ('deci
', 203), ('puedes', 199), ('puede', 195), ('vale', 194), ('dónde', 194), ('claro',
88), ('cosas', 187), ('noche', 186), ('espera', 182), ('parece', 182), ('usted', 18
), ('gente', 181), ('entonces', 180), ('mamá', 176), ('serio', 172), ('ahí', 171),
'oye', 171), ('amigo', 170), ('mira', 170), ('vas', 169), ('papá', 163), ('tal', 16
), ('quiere', 154), ('hombre', 153), ('pasa', 153), ('vaya', 151), ('trabajo', 150)
 ('amor', 150), ('genial', 150), ('tipo', 149), ('hablar', 147), ('dijo', 144), ('t
ner', 141), ('quizá', 137), ('gran', 136), ('tío', 134), ('chica', 132), ('podría',
132), ('luego', 131), ('momento', 130), ('nadie', 130), ('oh', 129), ('mierda', 129
, ('mal', 125), ('buena', 123), ('hecho', 123), ('necesito', 122), ('pues', 122), (
buen', 122), ('mundo', 121), ('solo', 121), ('hoy', 120), ('contigo', 120), ('ven',
118), ('mismo', 118), ('viejo', 118), ('tres', 117)]
```

→ En el género de horror

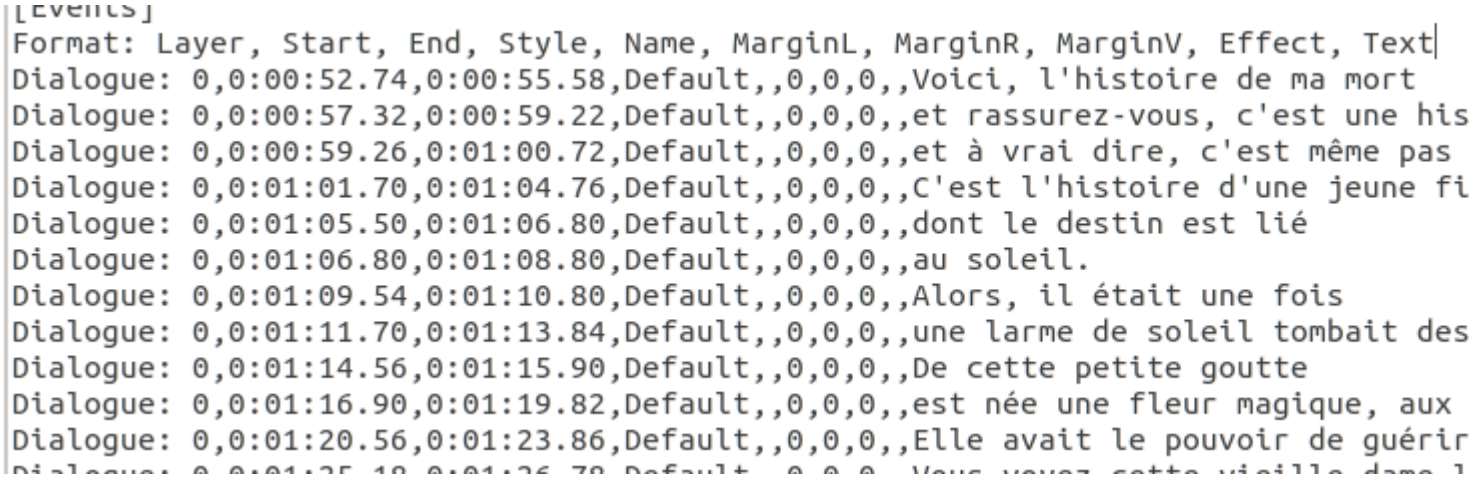


Noto que la lematización no fue buena, ya que verbos como querer, poder, ir, saber, o creer se encuentran conjugados de distintas maneras y son muy frecuentes en los 3 géneros, por lo que quizás valga la pena quitarlos. También creo que sería bueno quitar algunas palabras, como “aquí” “sí” o “bien” que siento que deberían ir en los stopwords.

- Para inglés realicé lo mismo, y al inspeccionar las palabras más frecuentes de los 3 géneros me doy cuenta de que igualmente verbos come “come” “know” “go” “get” “want” y “think” aparecen en alta frecuencia en los tres géneros y quizás considere añadirlas a los stopwords. Por su parte, también aparecen “ll” “re” y “don” que considero que son errores de la etapa de preprocesamiento, ya que provienen de la contracción de “will” “are” y “don't”, las cuales considero deberían estar definitivamente en los stopwords.

- Para el francés, las palabra “oui” y “non” aparecen entre las 10 palabras más frecuentes en los 3 géneros, y me sorprendió que estas palabras no estuvieran ya en la lista de stopwords. Se repite lo que sucede en español, que los verbos “ir” saber", “querer”, y “decir” aparecen conjugados de distintas maneras y con mucha frecuencia; creo que es necesario mejorar el lematizado de estos verbos o quizás añadirlas a los stopwords directamente. Otra palabra que creo que incluiré en stopwords es “quoi”, que se traduce como “qué", pero en el francés hablado suele usarse al final de las frases sin que tenga un significado o uso real... es también una de las palabras más frecuentes en los tres géneros.

- Otro fenómeno extraño que encontré en el corpus en francés, es que en el género de romance las palabras “default” y “dialogue” aparecen las dos 1090 veces, y se posicionan como las palabras más frecuentes del género... Esto me hizo sospechar y hacer una búsqueda manual, descubrí que los subtítulos en francés de la película “Tangled” se encontraban en un formato distinto al de los demás subtítulos:



cada línea de diálogo es marcada con el texto “Dialogue” y “Default”. Debido a esto, mejor descargué otros subtítulos en francés para esa película.

- Además, noto cosas interesantes como que:
→ “princesa”, "princess" y “princesse” son muuucho más frecuente en el género romántico que en los otros.
→ “hombre” y “homme” son más frecuentes en el género de acción, pero “man” es más frecuente en romance.
→ “mujer” “woman” y “femme” son más frecuente en el género de romance, pero en francés e inglés la diferencia es más notable que en español.
→ “padre”, “father” y “père” son notablemente más frecuentes en el género de acción, pero “papá”, “dad” y “papa” son más frecuentes en el de romance.
→ “madre”, “mother” y “mêre” son notablemente más frecuentes en el género de horror, pero “mamá” “mom” y “maman” son más frecuentes en romance.
→ “house” y “maison” son más frecuentes en horror, pero “casa” es igual de frecuente en horror y romance.
→ “morir” “die” y “mourir” son mucho más frecuentes en acción. Con “muerte”, “death” y “mort” sucede lo mismo.
→ “amar” “love” y “aimer” son mucho más frecuentes en romance. Con “amor”, “love” y “amour” sucede lo mismo.
→ “sangre” “blood” y “sang” son más o menos igual de frecuentes en acción y horror.

- En conclusión creo que la lematización salió mejor de lo que esperaba, pero no creo que sea suficiente... me causan particular ruido los verbos conjugados, y los sustantivos en plural... Por otro lado, parece que la frecuencia de palabras individuales se mantiene constante entre las 3 lenguas. De pronto no sé si hacer lematización con freeling, volver a hacer esto que hice y luego mejorar la lista de stopwords... creo que sí lo haré y espero que mejore la lematización.

17/Nov Sobre Instalar Freeling

- Aunque al principio tuve que dar varias vueltas para lograr utilizar Freeling en Python 3, al final resultó que no tenía que ser un proceso tan complicado.
- Primero solo se tiene que descargar el tarball de Freeling 4.0 estable desde los releases en Github, luego seguir estas instrucciones para compilar e instalar Freeling desde el tarball, después seguir las instrucciones para compilar una librería de Python 3 (las instrucciones vienen en el código fuente descargado en el tarball, y por ende también están en github) (para usarlas fue necesario instalar SWIG), tras instalarlo también es necesario explorar el ejemplo que viene incluido en el código fuente para entender cómo usar Freeling desde Python. Finalmente, tras entender cómo hacerlo, fue bastante sencillo escribir un lemmatizador que se ajustara a mis necesidades, y que lo pudiera usar para lemmatizar mis subtítulos.
- Cabe decir que Freeling es una librería y un conjunto de herramientas escritas en C++ y que se instalan en el sistema (típicamente en /usr/), por lo que no lo instalé directamente en el python environment que tengo para el proyecto (aunque creo que sería posible hacerlo, no le vi la necesidad). Por su parte, para usar la API con Python, solo es necesario compilar una librería y hacer una interfaz con SWIG (siguiendo las instrucciones mencionadas) y luego mover dicha librería e interfaz a donde tengo mis scripts para este proyecto, para así poder usar freeling. Solo hay que cuidar que en los scripts que usen freeling se indique la ruta de dónde en el sistema está instalado Freeling.
- Como referencia para todo el proceso, también leí esta guía en medium y esta guía en python, que ya están desactualizadas, pero igual me sirvieron para aclarar algunas dudas.

22/Nov Retomando el proyecto

- Tras unos días de no haber trabajado en el proyecto, lo retomé. Reestructuré la manera en que organizaba las carpetas para que estuvieran más organizadas y fuera más sencillo de explorar cuando envíe el proyecto.
- Por primera vez en este proyecto cree un cuaderno de jupyter para comenzar el minado de los tópicos.

22/Nov Sobre tópicos redundantes

- La primer pregunta que me surgió al comenzar a pensar en el minado de tópicos fue: ¿debería usar los subtítulos de todos los géneros para minar tópicos? ¿o debería dividirlo por géneros y entrenar un modelo por cada género? Al final me he decidido a hacer ambas opciones.
- Implementé el código para cargar los archivos de texto y convertirlos en índices para ser utilizados por la implementación de LDA de gensim.
- En la primer prueba que hice con la minería de tópicos me sorprendió que se hiciera tan rápido, aunque tiene sentido ya que los documentos son cortos y son pocos. El primer resultado no me agradó:

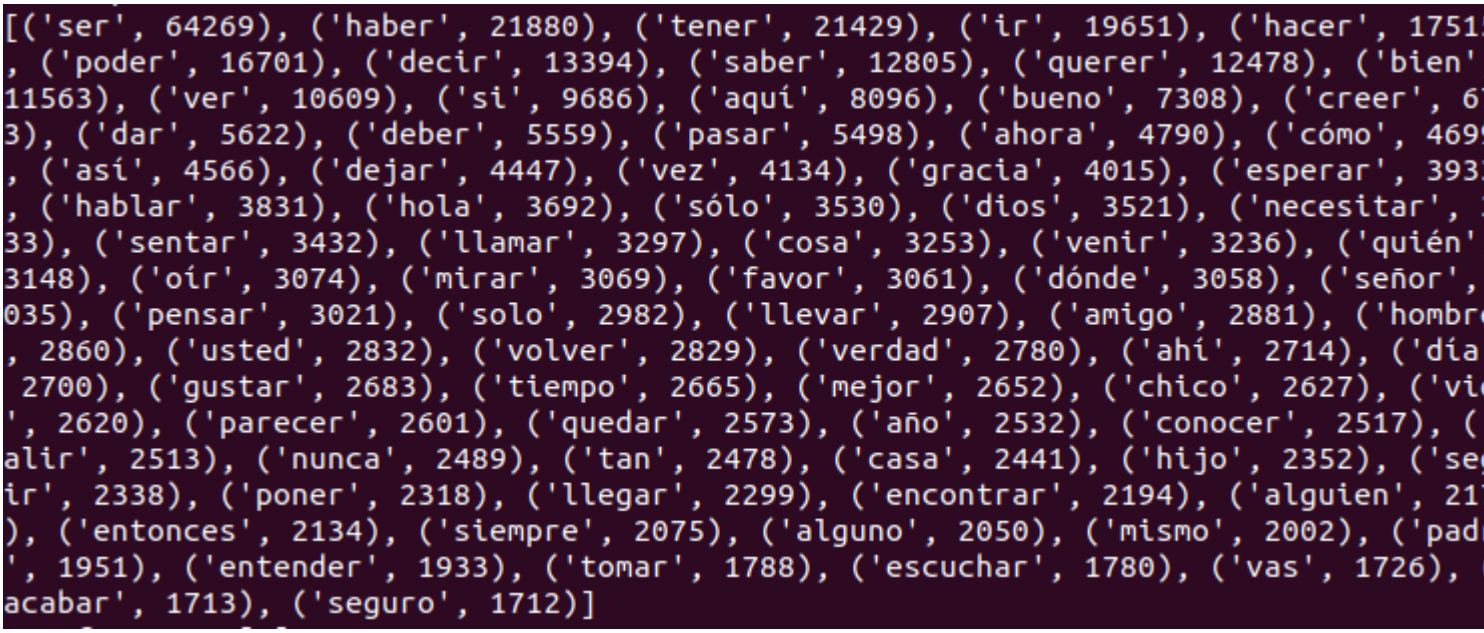
```
In [124]: topics_eng_accion.show_topics()

Out[124]: [(0,
  '0.018*"get" + 0.017*"know" + 0.015*"go" + 0.015*"can" + 0.01
014*"just" + 0.013*"yeah" + 0.012*"like" + 0.010*"will" + 0.016
 (1,
  '0.016*"get" + 0.015*"can" + 0.014*"will" + 0.012*"go" + 0.01
0.009*"just" + 0.008*"us" + 0.008*"well" + 0.008*"now" + 0.007*
 (2,
  '0.018*"get" + 0.016*"can" + 0.015*"go" + 0.014*"just" + 0.01
0.014*"will" + 0.011*"like" + 0.011*"right" + 0.011*"come" + 0.
),
 (3,
  '0.019*"will" + 0.013*"can" + 0.012*"get" + 0.011*"go" + 0.01
0.010*"know" + 0.009*"us" + 0.007*"oh" + 0.007*"now" + 0.007*"g
 (4,
  '0.015*"get" + 0.013*"go" + 0.012*"can" + 0.011*"bourne" + 0.
+ 0.008*"come" + 0.008*"right" + 0.007*"just" + 0.007*"yeah" +
k"'),
```

- Hay muchos tópicos muy similares, y en general todos están compuestos de las mismas palabras.
- Tras reflexionar un poco recordé que cuando dejé el proyecto en suspenso hace unos días, ya no escribí en la bitácora que tenía que ser necesario el agrandar la lista de stopwords para que incluyera a las palabras que fueran muy frecuentes en mi corpus tras lematizar. Ya que sospechaba que las palabras muy frecuentes no aportarían nada y solo causarían ruido en la minería de tópicos. Así que me dispongo a re-limpiar el corpus, quitando las stopwords que crea conveniente. Espero que con esto sea suficiente para tener mejores tópicos.

23/Nov Sobre ampliar la lista de stopwords

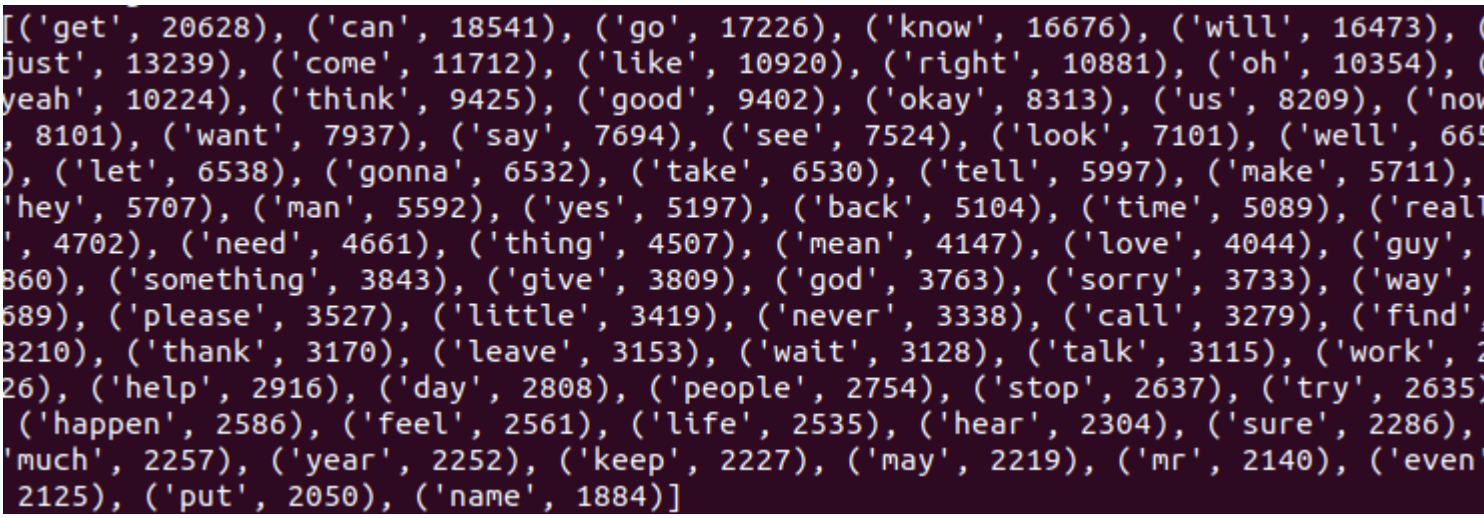
- Para decidir qué palabras añadir a las stopwords, usé un script que encuentra el conjunto de las 100 palabras más frecuentes de cada género, y tomo la intersección de esos conjuntos. Luego, con inspección manual de las frecuencias de las palabras, decido hasta dónde tomar para añadir a la lista de stopwords.

- Para español el resultado de las palabras más comunes es éste:



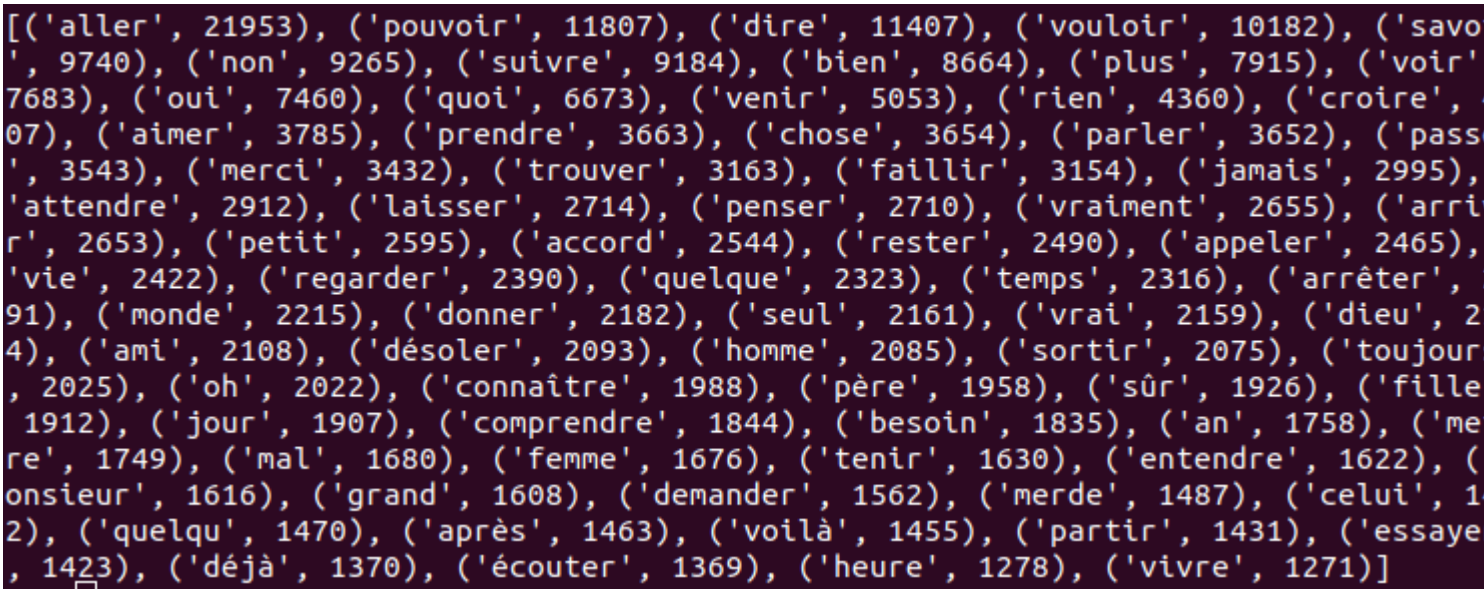
Se indica la frecuencia que tiene una palabra dada en todo el corpus. He decidido quitar todas las que tengan frecuencia mayor a ‘dios’ (3521) , y esas son 28 palabras.

- Para inglés el resultado es:



He decidido quitar todas las palabras que sean más frecuentes que ‘love' (4044) excepto ‘man’ y ‘time’. Son un total de 34 palabras.

- Y para francés el resultado es:



He decidido quitar todas las palabras con más frecuencia que petit (2595) excepto ‘aimer’. Son un total de 28 palabras.

- Noto que salvo la palabra ‘ser’ en español, las 3 distribuciones siguen más o menos la misma forma y contienen más o menos las mismas palabras; sobre todo verbos muy comúnes. Me llama la atención que el verbo ‘ser’ no estuviera ya en la lista de stopwords que estuve manejando; aunque se entiende ya que al parecer en la lista de stopwords que estuve usando en español no hay verbos en infinitivo.

- La principal diferencia entre las 3 distribuciones creo que es que en inglés las frecuencias se dividen de manera más equitativa.

23/nov - Los tópicos vuelven a ser redundantes

- Aunque haya quitado las stopwords previamente mencionadas, vuelve a haber tópicos redundantes. Aquí solo los primeros 4:

```
[(0,
  '0.007*"love" + 0.006*"time" + 0.005*"thank" + 0.005*"sorry" + 0.005*"little" + 0.005*"some
005*"great" + 0.004*"guy" + 0.004*"god"'),
 (1,
  '0.008*"man" + 0.007*"god" + 0.007*"guy" + 0.007*"fuck" + 0.006*"shit" + 0.006*"time" + 0.0
" + 0.005*"something" + 0.005*"love"'),
 (2,
  '0.010*"man" + 0.006*"time" + 0.006*"love" + 0.005*"guy" + 0.005*"please" + 0.004*"give" +
le" + 0.004*"day" + 0.004*"mr"'),
 (3,
  '0.008*"captain" + 0.007*"man" + 0.007*"ship" + 0.006*"sir" + 0.005*"time" + 0.005*"father
y" + 0.004*"king" + 0.004*"never"'),
```

- Se me ocurre que hay tres caminos a seguir:
1. Quitar aún más stopwords y dejar solo palabras que tengan una frecuencia similar
2. Dividir los documentos. Quizás lo anterior sucede porque tengo pocos documentos y en todos aparecen las palabras frecuentes. Quizás si los divido en documentos más pequeños ya no exista esta redundancia.
3. Dejar que LDA haga más iteraciones y más passes. Para esto me sería conveniente activar la opción de logguear el valor del desempeño a través de las iteraciones.

- De pronto me gustaría probar las 3 anteriores, y para eso creo que lo ideal sería preprocesar el corpus de nuevo pero sin quitar stopwords, para darme la libertad, más adelante, de experimentar con diferentes listas de stopwords para así ver cómo afectan al desempeño de LDA. También crear una función que segmente los documentos en pedazos del tamaño indicado (y creo que esto me servirá más adelante cuando haga el experimento de minar tópicos multilingües).

23/nov - 3/dic - Cuadernos de Jupyter

- Tras ponerme a seguir los caminos que indiqué en mi última entrada en la bitácora, he comenzado a crear los cuadernos de jupyter donde quedarán mejor documentadas mis ideas.