CAIM - Session 2

Programming with ElasticSearch

Santiago Arxé i Carbona

Bryan Leonardo Salto Salao

20/10/2021

1. Índices, filtros y *tokens*

El primer paso antes de realizar ningún tipo de análisis es obtener los ***tokens***, lo que podríamos describir como identificadores de palabras únicas en el texto. Esto se consigue aplicando un ***tokenizador***.

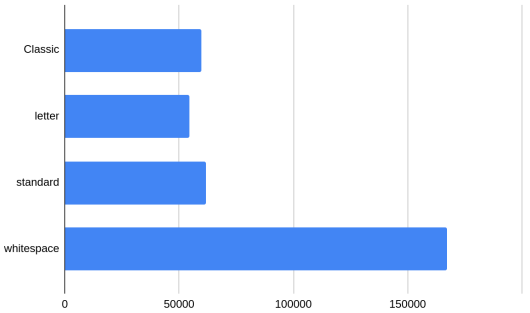

Figura 1: Cantidad de palabras diferentes obtenidas al aplicar cada uno de los *tokenizadores*

Como se puede observar en la Figura 1, la opción que crea más *tokens* es ***whitespace*** con 167063 palabras. De hecho, es predecible ya que consiste básicamente en crear **tokens** separados por espacios, por lo que números o guiones cuentan como partes de palabras, lo cual hará que “yes” y “yes,” sean considerados como palabras diferentes.

La opción ***standard***, la cual sí que tiene en cuenta los signos de puntuación. tiene un número mucho menor de palabras (61825). Aun así, sigue diferenciando palabras con guiones bajos, números o acrónimos.

La opción ***classic*** refleja una pequeña mejora respecto al caso anterior. Esto es debido a que esta opción está basada en la gramática inglesa. Con 59569 palabras, puede verse que los ejemplos mencionados anteriormente ya no se consideran palabras. Aun así hay fechas o algún número que sí, como por ejemplo: *“1899-1901”, “4/2/2/4228” o “4228”*.

El *tokenizador* más restrictivo es el ***letter***, con el que se consiguen 54455 palabras. Es normal ya que se basa en dividir en *tokens* cada vez que encuentre un carácter que no es una letra. Por lo que todos los casos anteriores ya no se tienen en cuenta.

Con la opción más restrictiva, se pasa a analizar diferentes tipos de filtros que pueden reducir aun más el número de “palabras” identificadas.

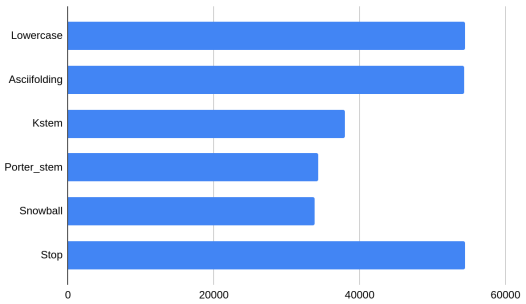

Figura 2: Cantidad de palabras diferentes obtenidas al aplicar cada uno de los filtros indicados

Como puede observarse en la Figura 2, los filtros ***asciifolding*** y ***stop*** son las que tienen un mayor número de palabras. Teniendo en cuenta que el tokenizador ***letter*** es muy restrictivo, muchas palabras que podían haber sido filtradas por el filtro ***asciifolding*** ya han sido eliminadas. ***Stop***, por

otro lado, elimina ciertas palabras con bajo nivel semántico, lo cual permite eliminar palabras con mucha frecuencia de aparición (pero no demasiadas palabras distintas).

Por otro lado, se aprecia como los filtros basados en ***stemming*** son los que han conseguido un menor número de palabras. En concreto ***snowball*** con 33783 palabras. Aquí puede verse como en un texto las palabras tienden a estar relacionadas y muchas de ellas a compartir una misma raíz.

Se procedió, por tanto, a combinar diversos filtros con tal de obtener el mejor resultado. Como puede observarse en la Figura 3, la combinación de filtros ***lowercase-asciifolding-stop-scowball*** es la que produce mejores resultados.

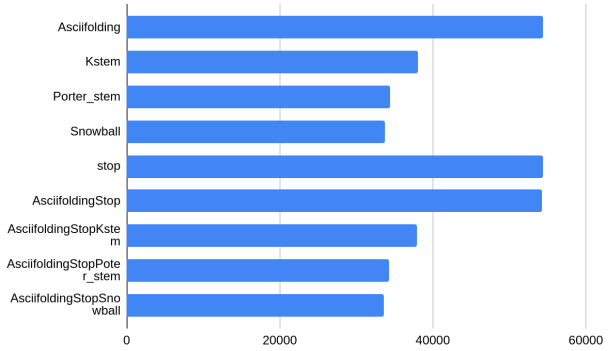

Figura 3: Cantidad de palabras diferentes obtenidas al aplicar cada uno de los filtros (siempre aplicando el filtro ***lowercase*** en primer lugar)

Algunas de las palabras más repetidas después de aplicar el filtro más restrictivo sobre la colección *arxiv_abs*:

- "model" con 78856 apariciones.
- "we" con 206553 apariciones.

También se ha añadido alguna otra opción como ***UAX_URL_Email tokenizer*** que identifica las *urls* como una sola palabra. Para ello solo se ha tenido que añadir el nombre de la opción en **IndexFilesPreprocess.py**.

2. Estudio de la "semejanza"

Una vez obtenida la mejor combinación de *token* y filtros (o al menos la más restrictiva), es hora de poner dicha combinación a prueba. Para ello, se pasará a analizar la semejanza entre diversos grupos de archivos con tal de poder comprender un poco más la medida de semejanza, lo que indica y cómo puede ayudar a discernir entre diversos tipos de textos.

Semejanza en un set

Un primer y rápido estudio que puede realizarse sobre la semejanza es el grado de semejanza, valga la redundancia, que hay dentro de diversos conjuntos. Esto permitirá, en cierto modo, comprobar lo similares que son los textos dentro de una misma categoría, así como ver lo importante que es el "estilo" y manera de escribir dentro de dichas categorías.

Para hacer este estudio, se ha analizado la semejanza (*similarity*) entre cinco archivos aleatorios dentro de tres categorías de textos. Como puede apreciarse en la Figura 4, el grupo de textos *rec* (que parece estar compuesto por correos electrónicos) es el que tiene una mayor semejanza, posiblemente por su naturaleza escueta y cordial. A su vez, el grupo de textos *comp* (el cual se

centra en diversos textos informáticos) tiene el menor valor de semejanza, seguramente debido a su carácter técnico y el uso de lenguaje mucho más específico.

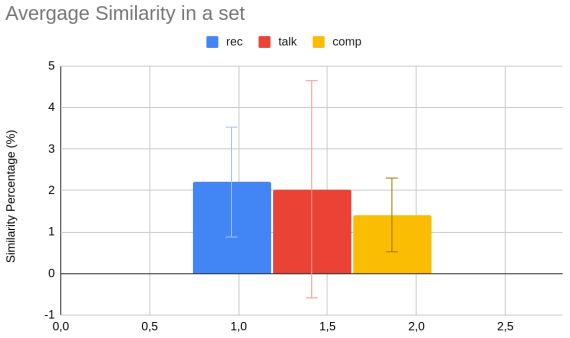

Figura 4: Semejanza media dentro de cada *set* y desviación estándar

Al mismo tiempo, es interesante observar cómo el grupo de textos con mayor variabilidad en su semejanza es *talk* (grupo de textos que parece mostrar extractos de debates). La gran variabilidad entre los temas, así como en el tono y contenido, hace que algunos textos se parezcan mucho entre sí y otros prácticamente nada.

Semejanza con otros *subsets*

Otro posible estudio es analizar la semejanza entre *subsets* después de indexar un *set*. Explicado con un pequeño ejemplo, lo que se hizo fue indexar el grupo de artículos *rec* y analizar la semejanza entre cinco parejas de artículos del *subset* ***autos***, cinco parejas formadas por un artículo de ***autos*** y otro de ***motorcycles***, y cinco parejas formadas por un artículo de ***autos*** y otro de ***baseball***.

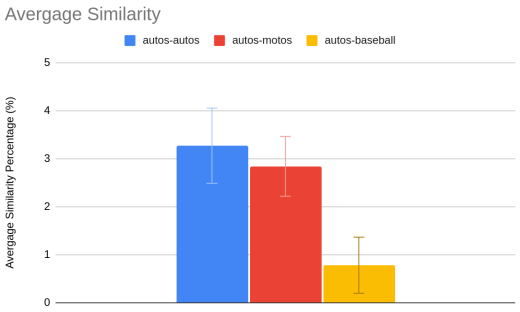

Figura 5: Semejanza entre artículos de diferentes *subsets*

La Figura 5 muestra de manera clara como las parejas formadas por artículos del mismo *subset* tienen una semejanza mayor a las parejas formadas por artículos de diferentes *subsets*. Además, cabe destacar que entre los artículos de diferentes *subsets*, las parejas formadas por artículos de coches y motos se asemejan mucho más a aquellas parejas formadas por artículos de coches y *baseball*. De este modo, podría llegar a concluirse que el *subset* ***motorcycles*** es mucho más cercano al *subset* ***autos*** que el *subset* ***baseball***.

Semejanza “entre los tuyos”

La medida de semejanza entre dos textos no es solo una medida de la cantidad de palabras iguales que tienen y la cantidad de veces que aparecen dichas palabras, sino que pone en valor cuánto de parecidos son los dos textos dentro del grupo que se está analizando (el índice al que pertenecen).

Para poner esto a prueba, se realizó un análisis de la semejanza de dos textos indexándolos en diversos conjuntos. Se tomaron las dos novelas de H. G. Wells (*Time Machine* y *War of the Worlds*) y se indexaron dentro del conjunto de ***novels***, así como dentro de los conjuntos de artículos ***talk*** (debate), ***comp*** (informática), ***rec*** (correos electrónicos) y ***sci*** (ciencia).

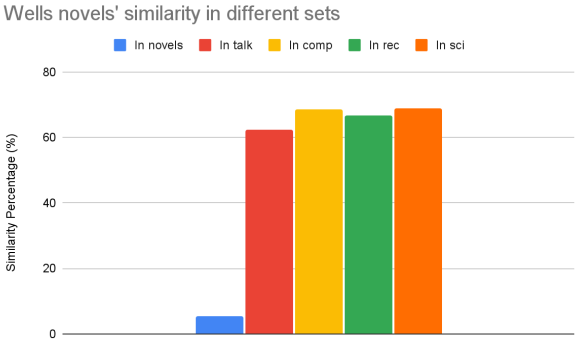

Figura 6: Semejanza entre dos novelas de H. G. Wells al ser indexadas en diferentes conjuntos

Como puede observarse de manera más que clara en la Figura 6, las novelas se “asemejan” mucho más entre ellas cuando se indexan junto a textos muy diferentes a ellas. Eso hace que sus palabras tengan un peso mucho mayor (al aparecer muchas de sus palabras en muy pocos textos dentro del índice) y su valor de semejanza crece de manera evidente.

Búsqueda de ***paths***

Como bien se explica en la documentación, todos los argumentos son *tokenizados*, lo cual haría imposible la búsqueda de un ***path*** con ***elastic search***. Para evitar esto, se añade un ***mapping*** al índice (a la hora de indexar en el *script* **IndexFilesPreprocess**) que almacenará los ***paths*** como propiedades en un diccionario.

Este importante y vital paso permitirá, más adelante, la detección de dichos ***paths*** mediante una búsqueda exacta (***exact match search***) cuando se vaya, por ejemplo, a calcular la semejanza entre dos archivos mediante la ejecución del *script* **TFIDFViewerB.**

Valoración

El código de la práctica era bastante claro y podía entenderse perfectamente. Dicho esto, el código a completar tenía un objetivo concreto y fuimos capaces de completarlo sin demasiadas dificultades. La implementación del cálculo de los ***tf-idf*** y de la ***cosine similarity*** nos ayudó a entender en gran medida el funcionamiento y significado de ambos cálculos.

En cuanto a los experimentos, fuimos capaces de analizar diversos aspectos de la semejanza entre archivos y nos permitió comprender en un nivel mayor su significado y los aspectos que entran en juego.