CAIM Lab, Session 2:
Programming with ElasticSearch

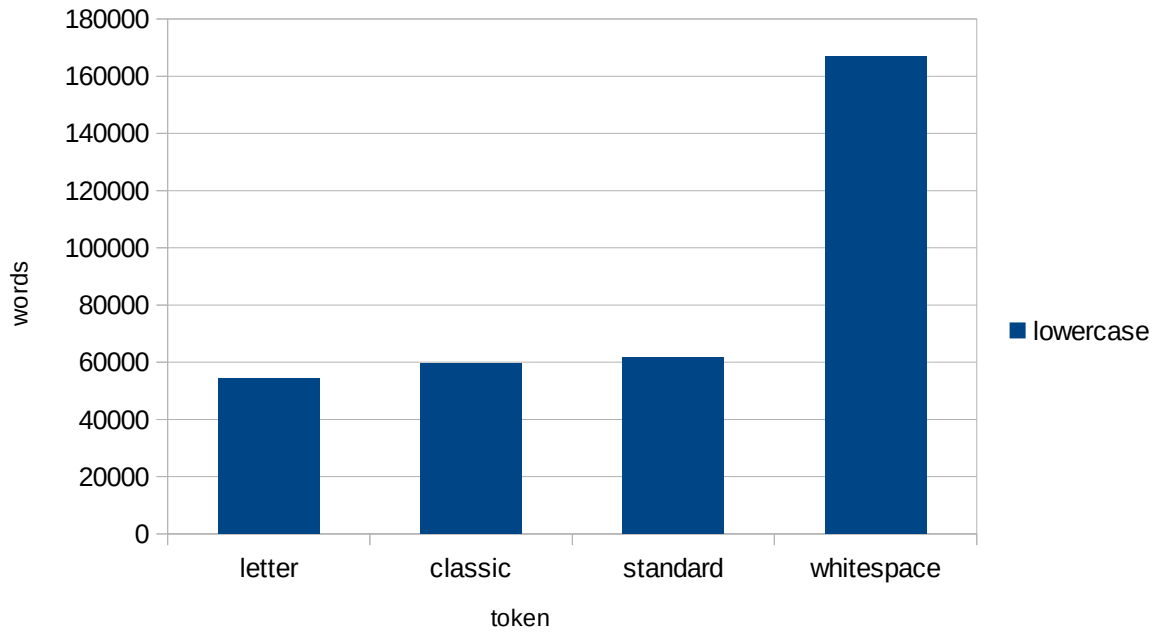
David Carballo Montalbán

David Sànchez Peña

Modifying ElasticSearch index behavior / The index reloaded

La primera part de la sessió consta de veure com afecten diferents tokenizers al nombre de tokens generats. Per a veure-ho, utilitzem el script IndexFilesPreprocess.py amb el nou flag –token. Les diferents opcions són les següents:

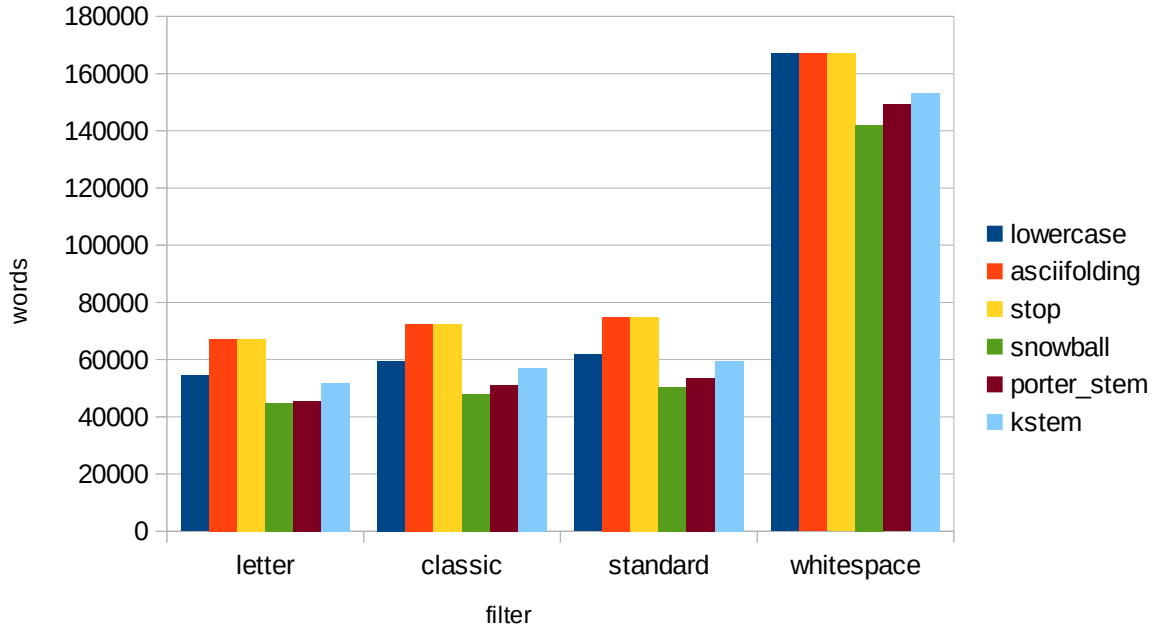
-Whitespace: Separa el text cada cop que troba un espai.
-Classic: Separa el text basant-se en la gramàtica. Compta amb heurístiques per tractar de forma especial acrònims, noms d'empreses i adreces electròniques.
-Standard: Separa el text en paraules seguint el Unicode Text Segmentation algorithm.
-Letter: Separa el text cada cop que troba un caràcter que no és una lletra.



Després d'executar tots els tokenizers, podem veure que letter és el més agressiu.

A continuació veiem la distribució dels filtres aplicats sobre cada token, provem els següents filtres amb el flag --filter:

-Lowercase: Normalitza el text a minúscules.
-Asciifolding: Transforma els caràcters que no són ASCII en el seu equivalent ASCII en el cas que existeixi.
-Stop: Elimina "stop words" de l'estil *in*, *a*, *the*...
-Stemming: Redueixen les paraules a la seva arrel. D'aquest tipus de filtre n'hi ha tres opcions: *snowball*, *porter stem* i *kstem*.



Després de fer l'experiment, veiem que els filtres més agressius són els que apliquen stemming. I d'entre aquests, el més agressiu és *Snowball*.

Computing tf-idf 's and cosine similarity

La segona part de la sessió consisteix en la comparació de documents completant el script *TFIDFViewer.py*. Aquest script utilitza el path i els ids del index per calcular els vectors tf-idf i poder calcular la similitut de coseno.

La primera funció a completar a sigut **toTFIDF**, hem calculat el term frequency i el inverse document frequency per obtenir cada pes i aconseguir el vector de pesos del document normalitzat, seguint aquests pasos:

Each weight is a product of two terms

$$w_{d,i} = tf_{d,i} \cdot idf_i.$$

The term frequency term tf is

$$tf_{d,i} = \frac{f_{d,i}}{\max_j f_{d,j}}, \qquad \text{where } f_{d,j} \text{ is the frequency of } t_j \text{ in } d.$$

And the inverse document frequency idf is

$$idf_i = \log_2 \frac{D}{df_i}, \qquad \text{where } D = \text{number of documents}$$

and df_i = number of documents that contain term t_i.

Després, la funció **normalize** que aplicava una normalització als pesos, on dividiem el pes de cada terme per el total de la suma dels quadrats d'aquests pesos.

La funció **cosine_similarity** calcula el producte escalar entre dos vectors de pesos. I finalment, la funció **print_term_weight_vector** que imprimia el vector.

Les implementacions d'aquestes funcions es poden veure al propi script.

A continuació, es mostren les comprovacions fetes amb el script per calcular la similaritat entre els diferents documents de 20_newsgroups.

Documents de alt.atheism vs sci.space:

Similaritat (alt.atheism/0000000 – sci.space/0014249) = 0,0213
Similaritat (alt.atheism/0000001 – sci.space/0014249) = 0,03168
Similaritat (alt.atheism/0000010 – sci.space/0014248) = 0,00977
Similaritat (alt.atheism/0000005 – sci.space/0014230) = 0,00879

Documents de sci.crypt entre si:

Similaritat (0011405 – 0011480) = 0,02237
Similaritat (0011405 – 0011418) = 0,01210
Similaritat (0011405 – 0011410) = 0,03999

Documents de alt.atheism entre si:

Similaritat (0000000 – 0000000) = 1
Similaritat (0000000 – 0000005) = 0,12252
Similaritat (0000000 – 0000002) = 0,04811
Similaritat (0000000 – 0000001) = 0,24181

Com podem observar en els arxius de la mateixa carpeta, obtenim valors més elevats en comparació a els arxius que es troben a diferents carpetes. Per exemple, en el cas de alt.atheism em obtingut una similaritat de fins a 0,2 i en canvi, quan hem comparat un d'aquests arxius amb un de la carpeta sci.space ens hem trobat amb una similaritat molt baixa de 0,00879.

També ens adonem que a vegades hi han arxius que encara que estiguin en carpetes similars o diferents tenen similaritats semblants.

Suposem que amb filtres i tokens més agressius, aquesta similaritat és més clara.