CAIM: Regla de Rocchio

Ana Mestre Borges, Omar López Rubio

8 de diciembre de 2017

1. Implementación

Para la implementación de la regla de Rocchio hemos seguido el modelo presentado en el documento. Esto es, ir añadiendo términos a la query inicial, tratada como un diccionario, fijándonos en los k documentos más relevantes, y ponderando la query antigua con el parámetro α y los términos más relevantes del TfIdf de los documentos con el parámetro β. No ha habido tampoco ninguna dificultad reseñable con el código más que entender completamente cómo funcionaba la regla de Rocchio y pelearnos con la sintaxis de Python.

2. Experimentación

En nuestra experimentación hemos utilizado diversas querys y hemos ido modificando los parámetros α, β, k, R y $Nrows$. Hemos utilizado el índice news, creado a partir del dataset de 20 news group, para hacer todas las consultas.

2.1. Experimento 1

Hemos utilizado como query toronto y $nhits = 5$, dentro del índice news. Los valores utilizados han sido $k = 10$, $beta = 1$, $alpha = 2$, $R = 4$ y $nrows = 10$.

La query formada por la regla de Rocchio ha sido:

['team,0.354784738754', 'gilmour,0.418618337088', 'detroit,0.512803628735', 'toronto,0.638633946746']

y se han encontrado 7 documentos.

2.2. Experimento 2

Utilizando la misma query, toronto, y cambiando los parámetros $k = 5$, $beta = 4$, $nrows = 5$ tenemos el siguiente resultado:

['team, 0.485367330583', 'detroit, 0.73134397422', 'gilmour, 0.602360541892', 'toronto, 0.821276635742']

con los mismos 7 documentos.

2.3. Experimento 3

Aplicando la misma query que en los otros experimentos, pero ahora con los valores $k = 10$, $beta = 1$, $alpha = 1$, $R = 5$, $nrows = 5$ tenemos:

['gilmour,0.230072299565', 'team,0.131753667235', 'maynard,0.0865609402202', 'toronto,0.951577745946', 'detroit,0.177966651782']

pero ahora 0 resultados.

2.4. Experimento 4

Ahora cambiamos la query por $science^2$ religion. Los valores en este caso serán $k = 10$, $beta = 1$, $alpha = 1$, $R = 4$, $nrows = 10$. La query final es:

['evolution,0.612384451846', 'science,0.267133242698', 'religion,0.336606100103', 'theory,0.361188562466']

retornando 32 documentos.

2.5. Experimento 5

Ahora modificaremos solo el parámetro α, de manera que $\alpha = 5$ La query en este caso ha sido:

['evolution,0.835122508291', 'science,1.05457543833', 'religion,0.802461632687', 'theory,0.495039363503']

con los mismos 32 documentos.

3. Conclusiones

Con la experimentación hemos visto qué efecto tiene cada parámetro:

3.1. *nRounds*

No hemos apreciado un cambio demasiado alto cuando nRounds tiende a aumentar, simplemente va dándole importancia a los otros valores porque aumenta el número de iteraciones.

3.2. R

En el caso de R, cuanto más alto era el valor menos documentos íbamos adquiriendo, como se puede ver en el experimento uno y dos. En otras palabras, adquiríamos una *precision* mayor pero iba disminuyendo el *recall* (menos documentos más relevantes pero muchos falsos negativos).

3.3. α y β

Tenemos que α hace que la query original del usuario mantenga su peso en las diversas iteraciones de la regla, mientras que β hace justo lo contrario, pondera la query formada por los TfIdf. Esto se puede apreciar en el experimento cinco, ya que al variar ese parámetro tenemos que la ponderación del usuario $science^2$ se mantiene más relevante que en el experimento cuatro.

3.4. k

En este caso, k nos indica el número de documentos para calcular su TfIdf, por lo que al aumentar este valor obtenemos más documentos, muchos de ellos poco relevantes; por lo que aumenta el número de falsos positivos. Es decir, aumenta el *recall* y disminuye la *precision*.