ACTO1 – SAR

(11/04/2016)

Apellidos y Nombre:……

(IMPORTANTE: todos los cálculos se mostrarán truncados a dos decimales)

1) ¿Cómo se construiría el índice **permuterm** de la palabra "placa"? Explica el mecanismo de búsqueda para la wilcard query "pl*ca". **(0,5 puntos)**

El índice permuterm para el término prisa se construiría con las diferentes rotaciones del término:
placa$
laca$p
aca$pl
ca$pla
a$plac
$placa

Y la búsqueda que se realiza es: ca$pl*
siguiendo la regla: Para buscar X*Y → buscar Y$X*

2) En una colección de test para una consulta tenemos 9 documentos relevantes . Entre los 10 documentos devueltos sólo 6 son relevantes ocupando las posiciones 1,3,4,6,9,10. **(1 punto)**

Se pide:

a) Calcula la eficacia del sistema sin tener en cuenta el orden de los documentos en términos de Precisión, Recall, F-medida con β=1 (No se puntuarán las respuestas que consistan únicamente en el valor resultante)

Precisión= 6/10 = 0,6 **Recall=** 6/9 = 0,66

F-medida= (2x0,6x0,66)/(0,6+0,66)=0,79/1.26=0,62

b) Completa las Tablas de Precision y Recall Reales (expresando la operación de división realizada y el resultado en decimales, p.e. 2/3 = 0,66) e Interpoladas.

Tabla Precision&Recall Reales

	1	2	3	4	5	6	7	8	9	10
Relevante	si		si	si		si			si	si
Precisión	1 1/1	0,5 1/2	0,66 2/3	0,75 3/4	0,6 3/5	0,66 4/6	0,57 4/7	0,5 4/8	0,55 5/9	0,6 6/10
Recall	0,11 1/9	0,11 1/9	0,22 2/9	0,33 3/9	0,33 3/9	0,44 4/9	0,44 4/9	0,44 4/9	0,55 5/9	0,66 6/9

Tabla Precision&Recall Interpoladas

Precisión	1	1	0,75	0,75	0,66	0,6	0,6	0	0	0	0
Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

3) Considerando los cuatro documentos siguientes:
Doc1: ¿Qué es la vida? Un frenesí.
Doc2: ¿Qué es la vida? Una ilusión, una sombra, una ficción; y el mayor bien es pequeño;
Doc3: que toda la vida es sueño,
Doc4: y los sueños, sueños son

Considerando que se hace stemming, y que las palabras "sueño" y "sueños" corresponden al mismo término (que denotaremos como **sueño**): **(1 punto)**

a) Completa la tabla tomando como términos **vida** y **sueño**, usando tf-idf (para el cálculo se toma log-pesado).

Término	Doc1	Doc2	Doc3	Doc4	df	idf	tf.idf (D1)	Norm(D1)	tf.idf (D2)	Norm(D2)	tf.idf (D3)	Norm(D3)
Vida	1	1	1	0	3	0,12	0,12	1	0,12	1	0,12	0,37
Sueño	0	0	1	2	2	0,3	0	0	0	0	0,3	0,92

b) Calcula la distancia coseno entre Doc1 y Doc2, y entre Doc2 y Doc3, con un esquema de pesado ltc (log-pesado, idf y coseno normalizado).

cos (Doc1 , Doc2) = 1 x 1=1

cos (Doc2 , Doc3) = 1 x 0.37 + 0 x 0.92=0.37

c) Calcula la distancia de Jaccard entre Doc1 y Doc2, y entre Doc2 y Doc3.

Jaccard (Doc1 , Doc2) = 1/1=1 **Jaccard (Doc2 , Doc3) = 1/2= 0,5**

4) Calcula la Distancia de Levenshtein entre las siguientes palabras, considerando que el coste de la operación Borrado es 1, Inserción es 1, y Sustitución es **1**. Utiliza la cuadrícula para representar los costes acumulados. La cuadrícula tienen un tamaño fijo, que no tiene por qué ajustarse exactamente al espacio que necesitáis utilizar. **(0,5 puntos)**

D(cansa, cantada) =3

a	5	4	3	2	2	1	2	3
s	4	3	2	1	1	2	3	4
n	3	2	1	0	1	2	3	4
a	2	1	0	1	2	3	4	5
c	1	0	1	2	3	4	5	6
#	0	1	2	3	4	5	6	7
	#	**c**	**a**	**n**	**t**	**a**	**d**	**a**