CS547: Information Retrieval **Spring 2016**

Homework #1

Solutions Compiled from
Aravind Balakrishnan and Mariheida Córdova Sánchez

1 Problem 1

Content Based Filtering - The query is stable and the collection changes. For example, the user may be interested in a specific query item "49ers Team" or a subject matter like "Deep Learning Techniques". When new content arrives, the Information Retrieval System needs to asses if the new content is "interesting and relevant" and present it to the user. In this scenario, the query is fixed but the collection changes unlike the AdHoc way of doing queries.

2 Problem 2

If we do not pre-process html documents based on structured properties, a query that uses common html tags would retrieve very poor results. For example, a query like *small blue mark in head*. The term *small* is used frequently in html documents to format text. The term *blue* can be used in html documents to color the text in a blue color. The term *mark* is used to highlight text. The term *head* is used as a tag in almost all html documents. If we do not preprocess these html documents and remove these tags, we could end up matching this query with an html document, which it should not be.

3 Problem 3

$$Precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN}$$

- **Controlled vocabulary:**

  With controlled vocabulary, the number of retrieved documents can go up. In other words, number of TP, FP or both can go up. Precision can either go up or down depending upon the situation. But, recall is monotonically increasing as the number of retrieved documents increases.

- **Stopwords:** Under most circumstances the precision is expected to increase because of the significant decrease in the number of false positives. Precision can go down only in cases where query is entirely composed of stop words (e.g., "to be or not to be"). In case of recall the false negatives can increase. Therefore in most cases, the recall is expected to remain the same but can go lower.

- **Stemming:** With stemming, the number of retrieved documents increases because morphological variants of a base word will be considered the same. Therefore, there is an increase in the number of true positives, false positives or both. Precision can either go up or down depending upon the situation. But, recall is monotonically increasing as the number of retrieved documents increases.

- **Phrases in single terms:** With phrases as single terms, the number of retrieved documents can go lower. In other words, number of TP or FP can go lower. Hence, the precision can either increase/decrease depending upon the query. But, FN can increase because of low retrieval. Therefore, in most cases recall is expected to go lower.

4 Problem 4

A balanced F-measure might not be a good "single number" metric for an end user because two different users might want results with different precision and recall. Maybe some users prefer to get all results that match (high recall at the expense of a low precision), while others will not be content with false negatives (high precision at the expense of recall). There are also various ways of getting the same F-measure, by varying the precision and recall. Two different systems with the same F-measure might have very different results. This is why F-measure might not always work well, depending on the system and the user needs.

5 Problem 5

5.1 Inverted index

Term	Frequency	Documents
Purdue	3	1, 3, 4
Information	2	1, 4
Retrieval	4	1, 2, 3, 4

Table 1: Inverted index

5.2 TF-IDF cosine similarity

Assumptions made: There is a total of 4 documents. Using lnc.ltc for the calculations. The TF uses a logarithmic weight (i.e. $tf = \log(tf\_raw) + 1$). When tf_raw is 0, the tf_weight is also 0. IDF is only used for the query, and it is $idf = \log(\frac{N}{df})$. In both cases (query and document, the normalization factor $\frac{1}{\sqrt{wt_1^2+wt_2^2+...+wt_j^2}}$ is used.

	tf-raw	tf-weight	df	idf	weight	normalized
Purdue	1	1	3	0.1249387366	0.1249387366	0.383332889
Information	1	1	2	0.3010299957	0.3010299957	0.9236102513
Retrieval	0	0	4	0	0	0

Table 2: tf-idf for query "purdue information"

tf-idf vector for query: (0.3833, 0.9236, 0)

	tf-raw	tf-weight	normalized
Purdue	2	1.301029996	0.6609608922
Information	0	0	0
Retrieval	3	1.477121255	0.7504203482

Table 3: tf-idf for document 3

tf-idf vector for document 3: (0.6610, 0, 0.7504)

Calculating the dot product of the query vector (results from table 2 and the document vector (results from table 3, the tf-idf cosine similarity of query and document 3 = $0.3833 \times 0.6610 + 0.9236 \times 0 + 0 \times 0.7504$
tf-idf cosine sim = 0.253

5.3 Which document is most likely about this course?

Let us assume that the Query is *"Information Retrieval"*.

	Query	Doc1	Doc2	Doc3	Doc4
Purdue	0	4	0	2	1
Information	1	3	0	0	3
Retrieval	1	1	3	3	3

We will calculate the cosine similarity between the query and each of the documents. We will use the nnc.nnc mechanism to calculate this.

Query and Doc	Cosine Similarity
Query and Doc1	$\frac{0*4+1*3+1*1}{\sqrt{0^2+1^2+1^2}*\sqrt{4^2+3^2+1^2}} = \frac{4}{\sqrt{2}\sqrt{26}} = \mathbf{0.5547}$
Query and Doc2	$\frac{0*0+1*0+1*3}{\sqrt{0^2+1^2+1^2}*\sqrt{3^2}} = \frac{3}{\sqrt{2}\sqrt{9}} = \mathbf{0.707}$
Query and Doc3	$\frac{0*2+1*0+1*3}{\sqrt{0^2+1^2+1^2}*\sqrt{2^2+3^2}} = \frac{3}{\sqrt{2}\sqrt{13}} = \mathbf{0.588}$
Query and Doc4	$\frac{0*2+1*3+1*3}{\sqrt{0^2+1^2+1^2}*\sqrt{1^2+3^2+3^2}} = \frac{6}{\sqrt{2}\sqrt{19}} = \mathbf{0.9733}$

Since the cosine similarity between the query and Doc4 is the highest, **Doc4** is the document that is most likely about this course.