A	$R=0$	$R=1$	**B**	$R=0$	$R=1$
$\hat{R}=0$	969	21	$\hat{R}=0$	957	23
$\hat{R}=1$	3	7	$\hat{R}=1$	15	5

Table 1: Contingency table for the two search engines.

Problem 1 (What comes first). Consider a repository of 1000 documents of which 28 are relevant to a user query. There are two search engines A and B. Search engine A returns 10 documents and search engine B returns 20 documents. An expert says that 7 of the 10 documents returned by A are relevant while 5 of the 20 returned by B are relevant.

(a) Calculate the precision, recall and accuracy metrics of the search engines.

(b) Which search engine is the winner in terms of its i) accuracy and ii) recall?

(c) Accuracy is not the best metric for IR evaluation. Briefly justify this statement based on the previous results.

(d) Let the order of relevant documents in the corpus be 1 (most relevant), 2, ..., 28 (least relevant). We mark the documents retrieved by seach engines A and B, by relevance and **X** if not relevant and get the following results.

A: 1 **X** **X** 4 2 **X** 3 6 7 5

B: 1 3 2 4 5 **X** .. (13 irrelevant documents) .. **X**

Which search engine do you think is a better performer for the top 5 results. Why?

Solution.

(a) The metrics can be obtained from the contingency tables 1 for the two search engines, where the actual relevant documents are labelled $R=1$ and the documents the search engine tags as relevant is labelled as $\hat{R}=1$.

Precision is the fraction of relevant documents over all the documents that the search engine retrieves. So,

$$Precision_A = \frac{7}{10} = 70 \tag{1.1}$$

$$Precision_B = \frac{5}{20} = 25 \tag{1.2}$$

Recall is the fraction of the relevant retrieved documents over all relevant documents in the corpus. So,

$$Recall_A = \frac{7}{28} = 25 \tag{1.3}$$

$$Recall_B = \frac{5}{28} = 17.8 \tag{1.4}$$

Accuracy is the fraction of the sum of relevant retrieved documents and non-relevant unretrieved documents over the entire corpus. So,

$$Accuracy_A = \frac{7+969}{1000} = 97.6 \tag{1.5}$$

$$Accuracy_B = \frac{5+957}{1000} = 96.2 \tag{1.6}$$

(b) Search engine A in both cases.

(c) In case of information retrival problems, almost always, the data classes are skewed with the majority of documents being irrelevant. As in this example, out of a 1000 docuemnt corpus, only 28 are relevant. As a result, while a system may have high accuracy, it may be suffereing from high false positives.

Considering the metrics of search engines A and B, we see that they have comparable accuracies. However, a user using system B gets more false positive results (15) than user A (3) which is not desireable in a good retrieval system. Comparing the recall and precision scores we see that A is a far better performer than B. Recall and precision scores evaluate important aspects of IR - how many relevant documents could be retrieved and what percentage was false positives.

(d) Here we see that the top five results in A have 3 relevant documents whereas B returns the top 5 relevant documents in almost the same order of relevance. Hence, intuitively, B performs beter. (*Later we will learn about metrics like precision@k to evaluate systems more strictly.*)

Problem 2 (Et tu, Brute?). Assume that you have created an inverted index of Shakespeare's Collected Works. The terms *Brutus*, *Ceasar* and *Calpurnia* have posting lists of sizes x, y and z, respectively.

a What is the time complexity for the querying the keywords `"Brutus"` AND `"Calipurnia"`? When is this achieved?

b What is the time complexity for querying `"Brutus"` OR `"Calpurnia"`?

c Make the following queries on Google (keeping uppercases intact) and report the number of estimated search results.

(a) Caesar

(b) Brutus

(c) Caesar AND Brutus

(d) Caesar OR Brutus

(e) Brutus AND Caesar

(i) Do the numbers in the first four queries follow Boolean logic? Briefly justify.

(ii) Do the numbers in the third and fifth query follow Boolean logic? Briefly discuss, with an example, why word order should affect search results.

Solution.

a The time complexity for quering is linear in the size of the postings list of both keywords, *i.e.,* $O(x + z)$.

It is achieved only when the postings list is arranged in increasing order on the document IDs. As a result, the intersection of the two posting lists (the desired result for this case) is obtained linearly traversing each list. In practice, we can avoid complete traversal of the longer list if we reach a document ID in the longer list greater than or equal to the last document ID in the shorter list.

b Here the time complexity is a tight bound over the length of the postings lists, $\Theta(x + z)$. Since, unlike AND queries, we always have to traverse the entire list.

c The order of the estimated search results are:

a Caesar: 598×10^6

b Brutus: 29×10^6

c Caesar AND Brutus: 3.6×10^6

d Caesar OR Brutus: 548×10^6

e Brutus AND Caesar: 3.8×10^6

(i) The number of search results on the conjunction query (Caesar AND Brutus) does satisfy boolean logic: the number of documents containing both terms cannot be more than the minimum of the document frequencies of the search keywords.

The number of search results on the disjuction query (Caesar OR Brutus) does not satisfy Boolean logic. The number of relevant documents is atleast equal to the maximum of the document frequencies of the search keywords and upper bounded by the sum of the document frequencies of the search keywords.

Here document frequency is another term for the length of postings list of a keyword.

(ii) Both queries individually follow Boolean logic (the results of the AND queries are less than the single term query results). Also, following Boolean logic, order of conjunctions should not affect the final result since AND operation is commutative. However we get more results in (c) an order of 10^5 than in (e).

Though not apparent in this query, word order can change the meaning of the search queries. Consider the two search queries `sky blue` and `blue sky` where the user might have different intents. While the first refers to a color shade in general, the second query refers to the color of the sky.

Problem 3 (Indexing and query processing).

(a) Create an inverted index for the corpus in Table 2. Include documents ids and frequencies(no need for positions). [1]

(b) Process the following 2 queries using TAAT (term-at-a-time) query processing and return top-1 result:
Q1: apple stores in paris.
Q2: apple songs.

i Are both answers relevant to the queries? What's the problem?

ii Briefly, propose a solution.

document id	**text**
d_1	.....With an apple I will astonish Paris.....
d_2	...I am still a very big fan of Fiona Apple....
d_3	....I love making apple strudel.....
d_4	....Apple has done a great job on the IPod....

Table 2

Solution.

(a) appl — d1,1 d2,1 d3,1 d4,1
astonish — d1,1
big — d2,1
done — d3,1

[1] i) Disregard stop words. ii) Stem words using the Porter Stemmer`https://text-processing.com/demo/stem/`

fan — d2,1
fiona — d2,1
great — d3,1
ipod — d3,1
job — d3,1
love — d3,1
make — d3,1
pari — d1,1
strudel — d3,1

(b) Q1: apple stores in paris.
Top-1 result is d1.
Accumulators: d1:2,d2:1,d3:1,d4:1.
Q2: apple songs.
Top-1 is any document picked randomly(to break ties).
Accumulators: d1:1,d2:1,d3:1,d4:1.

i no. Ambiguity, 'apple' according to this corpus can mean the company, the painting, a singer, or a fruit.

ii Any of these are acceptable: increase k to allow more aspects; diversifying results to allow novelty; query expansion.

Albert is making sweets for his class's Christmas party. To impress his peers, he wants to search for the best recipes. As we all have sweet tooth and trust Albert's input (query), we want to help him find the best recipes by ranking them according to his query.

Problem 4 (BM25).

DocID	Document
d_9	life is a piece of cake with coffee
d_{10}	best chrismast coffee cake with nuts is this coffee cake with nuts recipe
d_{11}	sour cream coffee cake
d_{12}	carrotcake and wul nuts

Table 3

Using Okapi BM25, rank the documents in the table 3 by following the steps below. Albert's query is *coffee cake with nuts*

(a) Calculate the BM25 scores for documents and write the acquired ranking. Use the given $IDF(t)$ from the table 4 (Note that the IDFs are calculated from a big text corpora for generalization). Use $k_1 = 1.5$ and $b = 0.75$.

$$\text{score}(d, q) = \sum_{t \in q} \frac{(k_1 + 1) tf_{t,d}}{k_1 \left((1 - b) + b \frac{|d|}{\text{avgdl}} \right) + tf_{t,d}} \cdot \text{IDF}(t)$$

(b) This time, first remove the stop words={a, and, best, is, of, this, with} from the documents (and the query) and redo the steps in part a.

(c) What is the effect of stopword removing? (**briefly** explain)

Term	IDF
coffee	3.6
cake	1.9
nuts	-0.8
with	2.4

Table 4

	coffee	cake	with	nuts	$\|d\|$
d_9	1	1	1	0	8
d_{10}	2	2	2	2	13
d_{11}	1	1	0	0	4
d_{12}	0	0	0	1	4

Table 5: Term frequencies

Solution.

(a) First find the term frequencies for each term. We have the term frequencies in the table 5. avgdl = 7.25 and we calculate the scores for each document

$$\begin{aligned}
S(q,d_9) &= \frac{2.5*1*3.6}{1.5*((1-0.75)+0.75\frac{8}{7.25})+1} + \frac{2.5*1*1.9}{2.61} + \frac{2.5*1*2.4}{2.61} + 0 = 7.54 \\
S(q,d_{10}) &= \frac{2.5*2*3.6}{1.5*((1-0.75)+0.75\frac{13}{7.25})+2} + \frac{2.5*2*1.9}{4.39} + \frac{2.5*2*2.4}{4.39} + \frac{2.5*2*-0.8}{4.39} = 8.08 \\
S(q,d_{11}) &= \frac{2.5*1*3.6}{1.99} + \frac{2.5*1*1.9}{1.99} + 0 + 0 = 6.89 \\
S(q,d_{12}) &= 0 + 0 + 0 + \frac{2.5*1*-0.8}{1.99} = -1
\end{aligned}$$

The ranking is: d_{10}, d_9, d_{11}, d_{12}.

$$\begin{aligned}
S(q,d_9) &= 4.49 \\
S(q,d_{10}) &= 8.08 \\
S(q,d_{11}) &= 6.89 \\
S(q,d_{12}) &= 3
\end{aligned}$$

The ranking is: d_{10}, d_{11}, d_9, d_{12}.

(b) Now we first remove the stop words and then calculate the term frequencies (table 6). avgdl = 4.75

	coffee	cake	nuts	$\|d\|$
d_9	1	1	0	4
d_{10}	2	2	2	8
d_{11}	1	1	0	4
d_{12}	0	0	1	3

Table 6: Term frequencies (stopwords removed)

$$
\begin{aligned}
S(q,d_9) &= \frac{2.5*1*3.6}{1.5*((1-0.75)+0.75\frac{4}{4.75})+1} + \frac{2.5*1*1.9}{2.32} + 0 = 5.92 \\
S(q,d_{10}) &= \frac{2.5*2*3.6}{1.5*((1-0.75)+0.75\frac{8}{4.75})+2} + \frac{2.5*2*1.9}{4.26} + \frac{2.5*2*-0.8}{4.26} = 5.5 \\
S(q,d_{11}) &= \frac{2.5*1*3.6}{2.32} + \frac{2.5*1*1.9}{2.32} + 0 = 5.92 \\
S(q,d_{12}) &= 0 + 0 + \frac{2.5*1*-0.8}{2.08} = -0.95
\end{aligned}
$$

Ranking: d_9 and d_{11}, d_{10}, d_{12}.

$$
\begin{aligned}
S(q,d_9) &= 5.92 \\
S(q,d_{10}) &= 9.25 \\
S(q,d_{11}) &= 5.92 \\
S(q,d_{12}) &= 2.87
\end{aligned}
$$

Ranking: d_{10}, d_9 and d_{11}, d_{12}.

(c) In this case, removing the stopwords reduces the documents' length and raises them in the ranking.

Problem 5 (SLM).
Suppose we want to search the following collection of christmas cookie recipes. Assume that the numbers in the table indicate raw term frequencies.

	milk	coffee	ginger	sugar	raisins	cinnamon	flour	eggs	wulnut	apples
d_7	2	0	1	0	0	0	0	0	1	1
d_4	1	0	2	1	1	0	2	1	2	0
d_8	0	0	0	2	3	1	0	4	0	0
d_3	3	0	0	2	0	0	0	2	1	0
d_1	4	0	0	4	0	1	1	0	0	0
d_2	1	1	0	2	0	0	0	0	1	0
d_6	1	0	0	0	0	0	1	1	0	2
d_5	2	1	1	0	2	0	5	2	0	2

(a) Determine the top-3 documents including their query likelihoods for the queries

$$q_1 = \langle \text{ginger}, \text{sugar}, \text{raisins} \rangle \quad q_2 = \langle \text{milk}, \text{eggs}, \text{wulnut} \rangle$$

using the multinomial model (i.e., $P(q|d) = \prod_{t \in q} P(t|d)$) with MLE probabilities $P(t|d)$.

(b) Determine the top-3 documents using Jelinek-Mercer smoothing ($\lambda = 0.5$).

(c) Determine the top-3 documents using Dirichlet smoothing (for a suitable α)

Solution.

(a) We first calculate $P(q_1|d)$ for all documents.
$P(q_1|d_7) = P(ginger|d_7).P(sugar|d_7).P(raisins|d_7) = \frac{1}{5} \times 0 \times 0 = 0.$
$P(q_1|d_1) = P(ginger|d_1).P(sugar|d_1).P(raisins|d_1) = 0 \times \frac{4}{10} \times 0 = 0.$
Similarly we have, $P(q_1|d_2) = P(q_1|d_3) = P(q_1|d_5) = P(q_1|d_6) = P(q_1|d_7) = P(q_1|d_8) = 0.$
And $P(q_1|d_4) = \frac{2}{10} \cdot \frac{1}{10} \cdot \frac{1}{10} = 0.002.$
So for q_1 we have d_4 and the top-1 document.

Computing as above for q_2, the top-2 documents with their scores obtained are: $d_3(= \frac{3*2*1}{8*8*8} = 0.011)$ and $d_4(= 0.002)$.

(b) For q_1, the Jelinek-Mercer smoothing for document d_1 is obtained as,
$P(q_1|d_1) = \prod \left(\lambda \frac{tf(q_1^i,d_1)}{|d_1|} + (1-\lambda)\frac{tf(q_1^i,D)}{|D|}\right)$
where D is the concatenation of the 8 input documents $|D| = 68$, and $q_1^i \in \{ginger, sugar, raisins\}$.
Computing as above the top-3 documents (and their scores) are: $d_4(0.0016), d_8(0.001)$ and $d_5(0.00056)$.

Similarly, for $q_2 =< milk, eggs, wulnut >$ the top-3 documents are: $d_3(0.0057), d_7(0.003), d_4(0.0026)$.

(c) For q_1, the Dirichlet-Prior smoothing for document d_1 is $P(q_1|d_1) = \prod(\frac{tf(q_1^i,d_1)+\alpha\frac{tf(q_1^i,D)}{|D|}}{|d_1|+\alpha})$ with the legends as defined above. We select α as the average document length (= 8.5).
The top-3 documents are: $d_4(0.00164), d_8(0.001)$ and $d_7(0.00068)$.

With similar computations, the top-3 documents for q_2 are: $d_3(0.0058), d_7(0.0031)$ and $d_4(0.00256)$.

Problem 6 (Evaluate).
Finally we want to evaluate different ranking approaches (to help Albert choose the best method).

Consider documents assessed by experts (0 = irrelevant, 1 = somewhat relevant, 2 = very relevant) in table 7, where $d_i...d_j$ mean documents i, i+1, i+2, ..., j. Assume we used two different ranking methods to

DocId	$d_1...d_5$	$d_6...d_{10}$	$d_{11}...d_{20}$
Grade for query1	2	1	0
Grade for query2	1	2	0

Table 7

rank the documents for two different queries. And this is their results:
Ranking1(query1)= d_6, d_1, d_{11}, d_{12}, d_2, ...
Ranking2(query2)= d_{20}, d_9, d_{10}, d_3, d_4, ...
**Pay attention that the grades for query1 and query2 are different for each document.

(a) Calculate Precision@3 and Precision@5 for two ranking methods. Assume that grades 1 and 2 are mapped to "Relevant" and grade 0 is mapped to "Irrelevant".

(b) Calculate DCG@3 and DCG@5 for both ranking methods.

(c) Calculate NDCG@3 and NDCG@5.

Solution.

(a)

$$P@k = \frac{\#\text{relevant docs retrieved in top-k}}{k}$$

Ranking1:

$$P@3 = \frac{2}{3}$$

$$P@5 = \frac{3}{5}$$

Ranking2:

$$P@3 = \frac{2}{3}$$

$$P@5 = \frac{4}{5}$$

Ranking2 seems to be better, but we actually cannot compare the metrics because the queries are different.

(b)

$$DCG@k = \sum_{m=1}^{k} \frac{2^{grade(m)} - 1}{\log_2(1+m)}$$

$$NDCG@k = \frac{DCG@k}{IDCG@k}$$

Ranking1:

$$DCG@3 = \frac{2^1 - 1}{\log_2(2)} + \frac{2^2 - 1}{\log_2(3)} + \frac{2^0 - 1}{\log_2(4)} = 2.89 - LN = 4.17 - log10 = 9.6$$

$$DCG@5 = DCG@3 + \frac{2^0 - 1}{\log_2(5)} + \frac{2^2 - 1}{\log_2(6)} = 4.05 - LN = 5.85 - log10 = 13.46$$

Ranking2:

$$DCG@3 = \frac{2^0 - 1}{\log_2(2)} + \frac{2^2 - 1}{\log_2(3)} + \frac{2^2 - 1}{\log_2(4)} = 3.39 - LN = 4.89 - log10 = 11.27$$

$$DCG@5 = DCG@3 \frac{2^1 - 1}{\log_2(5)} + \frac{2^1 - 1}{\log_2(6)} = 4.21 - LN = 6.07 - log10 = 13.98$$

(c) Ranking1:
Ideal Ranking1(query1)= d_1, d_2, ..., d_5, d_6, ..., d_{10}, d_{11}, ..., d_{20}

$$IDCG@3 = \frac{2^2 - 1}{\log_2(2)} + \frac{2^2 - 1}{\log_2(3)} + \frac{2^2 - 1}{\log_2(4)} = 6.39 - LN = 9.22 - log10 = 21.24$$

$$NDCG@3 = \frac{2.89}{6.39} = 0.45$$

$$IDCG@5 = \frac{2^2 - 1}{\log_2(2)} + \frac{2^2 - 1}{\log_2(3)} + \frac{2^2 - 1}{\log_2(4)} + \frac{2^2 - 1}{\log_2(5)} + \frac{2^2 - 1}{\log_2(6)} = 8.84 - LN = 12.76 - log10 = 29.38$$

$$NDCG@5 = \frac{4.05}{8.84} = 0.458$$

Ranking2:
Ideal Ranking2(query2)= d_6, d_7, ..., d_{10}, d_1, ..., d_5, d_{11}, ..., d_{20}

$$IDCG@3 = 6.39 - LN = 9.22 - log10 = 21.24$$

$$NDCG@3 = \frac{3.39}{6.39} = 0.53$$

$$IDCG@5 = 8.84 - LN = 12.76 - log10 = 29.38$$

$$NDCG@5 = \frac{4.21}{8.84} = 0.47$$