2 Text preprocessing and indexing

Given a collection of text documents, you build two inverted indices:

- Index A -- no text preprocessing
- Index B -- stemming

Consider the total size (on disk) of **inverted lists** in each of these indices.
(Note, that the size of the vocabulary is **NOT** considered in this question).

In each of the following cases, how does the total size of inverted lists in index B compare to the total size of inverted lists in index A?

Please explain your answers in detail. Statements like "A < B" without explanation will result in 50% of the grade.

2.0p a The indices contain document ids.

2.0p b The indices contain term frequencies.

2.0p c The indices contain term positions.

3 Offline evaluation, metrics

For each of the following evaluation metrics, come up with one application/search scenario, where this metric is most suitable.

Explain, why it is most suitable for your application.

2.0p a Precision@1

2.0p b Full/Total recall

2.0p c ERR

4 Offline evaluation, test collections

Consider a test collection, where relevance judgments are created using depth-k pooling with standard IR ranking methods, such as VSM, QLM, BM25, LSI, and LDA. Consider also that a completely new set of ranking methods was developed after the test collection had been created, e.g., neural ranking methods.

2.0p a What problem may arise if you use the above test collection to perform offline evaluation of the new ranking methods?

1.0p b How can the test collection be modified to fix this problem?

1.0p c Is it possible to use the modified test collection to perform offline evaluation of standard IR ranking methods, such as BM25 and LSI? Explain your answer.

5 Term-based retrieval

You would like to use bi-gram language models for ranking.

1.0p a Give an equation to compute the Query Likelihood Model $p_{bi}(q \mid d)$ in this case. Express $p_{bi}(q \mid d)$ in terms of probabilities. Take into account that a query may contain more than one term.

2.0p b For each probability used in the above equation, explain, how it is calculated for document d.

6 Semantic retrieval

You use two low-rank approximations for LSI:

- LSI_1: $k = 1000$
- LSI_2: $k = 100$

2.0p Explain, why total recall is higher for LSI_2 compared to LSI_1?

7 Offline LTR

Consider the pairwise approach to offline Learning to Rank (LTR).

2.0p a In pairwise LTR, preferences between two documents are modeled.
Consider the following model that predicts preferences for each pair of documents: $f(d_i, d_j) = P(d_i \succ d_j)$.

What is a practical problem with this model?

1.0p b How can this problem be solved?

1.0p c Pairwise LTR (e.g., RankNet) has a smooth differentiable loss function and, thus, can be optimized using gradient descent. Still, there is a fundamental problem with the pairwise LTR approach. What is this fundamental problem?

2.0p d Give an example ranking that illustrates the above fundamental problem. Explain your example.

1.0p e How does LambdaRank solve this problem?

8 IR-user interaction

Consider using clicks for IR evaluation instead of relevance judgements.

2.0p a Give two pros of using clicks instead of relevance judgements.

2.0p b Give two cons of using clicks instead of relevance judgements.

2.0p c What information about user search behavior can be added to clicks to improve their quality/reliability? How should clicks be combined with this additional information?

9 Counterfactual LTR

Consider counterfactual learning to rank (LTR).

1.0p a Counterfactual LTR assumes a relation between a click on a document and the document's relevance. What is this assumption?

2.0p b Give two situations where this assumption does not hold.

10 Online evaluation

2.0p Explain, why A/B testing requires more user interactions than interleaving (you can also explain the other way around, i.e., why interleaving requires fewer user interactions).

11 Recommender systems

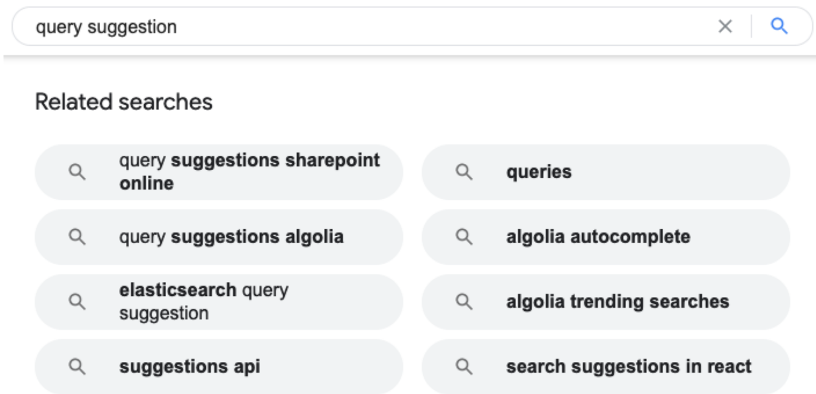
Consider the neighborhood-based approach to collaborative filtering. In this approach, a missing rating r_{ui} by user u for item i is calculated based on the k-nearest neighbors of user u. This is also know as a user-based approach, because we consider neighbors of the user u.

1.0p a Give an equation for calculating rating r_{ui} for item-based approach, where we consider k-nearest neighbors of item i.

2.0p b Given an item, explain, how to find its k-nearest neighbors.

12 Query suggestion

Query suggestion is a task, where, given a query q and a set of all possible queries $\mathcal{Q}$, one needs to rank $q_i \in \mathcal{Q}$ according to how likely a user is to submit q_i given that she submitted q. An example of *query suggestion* can be seen in the following figure:



Propose algorithms for *query suggestion*. The proposed algorithms should clearly explain how $q_i \in \mathcal{Q}$ are ranked with respect to q. For example, how the score for each q_i is calculated.

Note 1: Vague terms, such as "similar", "close", "higher", etc., will only be given 50% of the points. Instead, please use specific terms, such as "<name_of_similarity_function>", "distance of 2", "larger by 1", etc.

Note 2: The proposed algorithms should be implementable without any additional information. You can refer to any notion presented during the course without explaining it in detail, e.g., "inverted index", "MAP", "LSI", "LTR", etc.

1.0p a Propose a content-based *query suggestion* algorithm that only uses the content of q and $q_i \in \mathcal{Q}$.

2.0p b Suppose you have a query log which contains submitted queries (all those queries are in $\mathcal{Q}$), the order in which the queries are submitted, and the timestamp for each submitted query.

Propose an interaction-based *query suggestion* algorithm that only uses this query log (and does **NOT** use any content information).

2.0p **c** Suppose the query log also contains the returned search results for each submitted query and the corresponding click information.

Propose an interaction-based *query suggestion* algorithm that uses this click information (and still does **NOT** use any content information).