Exam on Information Retrieval (18-25/01/2022)

General note:

You can submit any time until 25th of January 2022, 1am. Until then, you can also reupload your solutions. Only the latest version will be evaluated.
Please upload your solution to the Final Exam/Submission folder.

Q1 (2)

What are the parts of an information retrieval system? How would you characterize them?

Q2 (2)

Which of the following statements are correct? You are allowed to tick either “true”, “false” or “it depends”. If you choose “false” or “it depends”, please justify your claim.

- “In scenarios where a user wants the results sorted by e.g. filename or date, you do not need to calculate the TF-IDF.”
- “The TF-IDF score of a document depends only on how often a term shows up in a document.”
- “You cannot combine stemming technology with lower-casing function. These technologies will get in each other’s way, and the result of the preprocessing would be useless.”
- “You cannot combine stemming technology with lemmatizing technology. These technologies will get in each other’s way; thus the result of the preprocessing would be useless.”

Q3 (4)

You can find three English texts by famous authors in the data folder: “The Picture of Dorian Gray” by Oscar Wilde; “Adventures of Huckleberry Finn” by Mark Twain, and “The hound of the Baskervilles” by Sir Arthur Doyle.

A friend wants to know how many different words each author used in each book. By “different words” he refers to the ground form of a word. To illustrate what he means, he uses the following paragraph as example:

> This is a sentence. There are many sentences.

This paragraph has 8 words, but only 6 different words:

("this", "be", "a", "sentence". "there", "be"*, "many", "sentence"*)
Words marked with * appear twice.

Please write a program that is able to calculate the number of different words in each book.

Additional remark: You can code file paths directly into your program. And you do not need to program a user interface. You are allowed to use any library that is available free of charge. The result will differ using different libraries, so don't worry if your routine delivers different results than your colleagues.

Please upload the program as Python or R script, you are allowed to use Jupyter Notebook or MarkDown syntax for your script.

Q4 (4)

A friend is troubled. She has hired a company to set up a webshop for her business selling spices from all over the world. The website looks nice; and customers can already pay for the products using a secured purchasing method. However, the search function in the webshop does not work as expected. The problem is that often there are false negatives.

You looked into the code, and found out that the code works as follows (in pseudo code):

```
// INDEXING PART
function index(document, document_path) {
      terms = get_terms_from_document(document)
      for (term in terms) {
            put_term_into_index(term, document_path)
      }
}

function get_terms_from_document(document) {
      content = pre_process(document)
      terms = split_content_into_words(content)
      return terms
}

function put_term_into_index(term, document_path) {
      // looks good here
      // the index is updated and saved
      // returns nothing
}
```

return может нужен??

dict = put_term_into_index(term, document_path)
return dict

поменять местами

```
function pre_process(document) {
      // Looks good here
      // The text is lower-cased and then stemmed
      // using a well-known library
      return preprocessed_document_as_simple_string
}

function split_content_into_words(content) {
      // Looks good here
      return a_set_of_terms
}

// SEARCHING PART
function search(querystring) {
      queried_terms = split_content_into_words(querystring)
      documents = find_documents_by_terms(queried_terms)
      ranked_documents = rank_documents(documents, querystring)
      return ranked_documents
}

function find_documents_by_terms(terms) {
      // Looks good here
      return a_list_of_document_paths
}

function rank_documents(documents, querystring) {
      // Looks good here
      return ranked_documents
}
```

добавить pre_process (потому что вдруг вводят с большой буквы, а у нас все с маленькой в словаре)

Can you see what the problem is? And how can you help fix the code?
Please note that “looks good here” means that the function does what it says and works without bugs.

Good luck!