INFORMATION RETRIEVAL(CSE508)

ASSIGNMENT 1

Group Member 1: Adarsh Singh Kushwah **RollNo:** MT21111
Group Member 2: Charisha Phirani **RollNo:** MT21117

Note: The supporting code is present in .pynb file

LIBRARIES USED:

1. **os**: for importing dataset folder from directory.
2. **nltk**: for performing operations like tokenization, stemming, detecting stop words etc.
3. **pandas**: for implementing dataFrames.
4. **string**: for working with text data present in dataset files.
5. **re**: Provides operations of detecting patterns with the help of regular expressions

PREPROCESSING

DATA UPLOADING

1. Used google.colab to import files from the device

DATA UNDERSTANDING

1. Read data using os.listdir("file location").
2. Checked the total count of files in the folder. The number of files in the folder are 1133.
3. A dictionary is created to store the filenames with their corresponding locations. The key contains the filenames and value contains the location of the file.

4. The key and value of the dictionary are taken and a dataframe is created for storing the filenames and their corresponding locations.

5. Lists are created for storing unique words and dataset corpus.

6. The dictionaries are created for storing posting lists, frequency of tems.

PRE-PROCESSING STEPS ON THE DATASET

1. Converted the complete text in all the files to lower case using .lower() function.

2. Then used .strip() to remove the blank spaces from all the files in the folder.

3. Used re.sub() to remove the punctuation marks from the text files.

4. Removed stopwords from the text file and performed tokenization, stemming and lemmatization.

5. In the dictionary created frequency of each term is stored as a value and the term is taken as a key. This will give the count of each term in different documents.

6. The terms whose frequency count is 1 are unique words and stored in a list named uniqueWords.

7. Filled the posting list and posting lists dictionary for every term in the file.

PRE-PROCESSING on the INPUT query

1. Convert the query to lower case.

2. Strip all the white spaces from the query.

3. Remove punctuation marks.

4. Perform tokenization and lemmatization.

METHODOLOGY

FUNCTION DEFINITIONS FOR OR, AND, NOT, AND NOT and OR NOT

AND FUNCTION IMPLEMENTATION

1. There is a list called output which stores the document names in which terms are matched. Initially the list is empty.

2. For the two posting lists termwise the documents are compared and the iterator is moved until one of the lists is exhausted.

3. If the document numbers are matched, the iterator moves forward putting the document number matched in the output list.

4. The number of comparisons are counted as the iterator moves and comparisons are done.

5. The output list storing the documents matched and the count of comparisons will be returned.

OR FUNCTION IMPLEMENTATION

1. There is a list called output which stores the document names in which terms are matched. Initially the list is empty.

2. The document numbers are matched and if the document is matched the document is added to the output list and the iterator for both the posting lists is moved forward.

3. If the document numbers don't match the smaller document number is added to the output list and the iterator skips that document number in that particular posting list.

4. The number of comparisons are counted as the iterator moves and comparisons are done.

5. The output list storing the documents matched and the count of comparisons will be returned.

NOT FUNCTION IMPLEMENTATION

1. The NOT function returns those documents which do not contain the terms or those not present in the posting list.

2. The document numbers present are matched with the document numbers present in the posting lists and the count for comparisons is increased.

3. Those document count which are not present in the posting list are added to the output list.

4. If all the document numbers are present in the posting lists, then output list will not contain any document so it will return 0.

ANDNOT FUNCTION IMPLEMENTATION

1. AND NOT function is the combination of AND and NOT performed together.

2. At first the NOT function is implemented on one of the posting lists using the NOT function.

3. After that the AND function implementation is applied on the posting list1 and **not posting list** obtained from not function and the output list contains the result of the combination. The comparisons are the total count of NOT and AND function implementation.

ORNOT FUNCTION IMPLEMENTATION

1. OR NOT function is the combination of OR and NOT performed together.

2. At first the NOT function is implemented on one of the posting lists using the NOT function.

3. After that the OR function implementation is applied on the posting list1 and **not posting list** obtained from not function and the output list contains the result of the combination. The comparisons are the total count of NOT and OR function implementation.

Some points to remember about other functions:

- There is a helper function defined which takes which operation is needed to perform on the posting lists.

- The output list returns the documents obtained after applying the functions and the totalComputations returns the number of comparisons after applying the functions.

As the input query by the user is passed, the preprocessing is done on the input query as well.

QUESTION 2

DATA UPLOADING

1. Used google.colab to import files from the device

Ans 2a)

Data Preprocessing

1. Converted the text in the files to lower case using .lower() function.

2. Using nltk library performed tokenization and generated tokens and removed stop words from the tokens generated.

3. Removed punctuation marks via re.sub library.

4. Removed blank spaces using .strip() function.

Ans 2b)

Positional Index Data Structure.

1. Created dictionary for storing file names along with index numbers in nameDictionary.

2. Created an empty list called uniqueWords for storing unique words in the dictionary.

3. There is an empty list created for storing the processed text called datasetCorpus.

4. Stored the position of words in the frequencyDictionary as values.

Ans 2c)
The input query is taken from the user and preprocessing is also done on the input query.

Preprocessing on input query.

1. Convert the query to lower case.

2. Strip all the white spaces from the query.

3. Remove punctuation marks.

4. Perform tokenization and lemmatization.

Implementing the positional indexing

1. If the number of tokens generated is 0 then the output will show 0 documents as the number of documents retrieved.
2. If the number of tokens generated is 1 then all documents will be searched for 1 word phrase or token generated and the document names will be displayed along with their Ids.
3. If the number of tokens generated is 2 then all documents will be searched for 2 word phrases or token generated and the document names will be displayed along with their Ids.
4. Similar steps will be followed for 3, 4, 5 tokens generated respectively.
5. If the number of tokens generated are more than 5, then the function will not be executed because the query length exceeds 5.
6. If there are random phrases provided by the user which are not present in the document, then invalid phrase message will be displayed.

ASSUMPTIONS TAKEN:

1. Removed single alphabet words from the files while preprocessing.
2. We have not removed 2 or more alphabet words which are not stopwords from the files which may effect the results of some input queries.