REPORT

Question 1.

Methodology

1. Loading of data files from “20 newsgroup” dataset.
2. Tokenization of the loaded data files.
3. Reading the document relevance score from the given file.
4. Storing triples of [file name, tokenized body, relevance score] in a list.

   Structure:
   [[file name1, tokenized body1, relevance score1], [file name2, tokenized body2, relevance score2] …]]
5. Lemmatization, Punctuation Removal, and Stop word removal from the tokenized body.
6. **Score Normalization**: Dividing each relevance score by the maximum score in the doc.
7. Counting term frequencies in each document for TF calculation.
8. Counting term wise TFs and IDFs.

$$\mathbf{TF\ (q, d) = 1 + \log(term\_frequency(q, d))}$$

$$\mathbf{IDF(q) = \log(num\_docs\ /\ (1+df(q))}$$

   TF scores computed are normalized by dividing with the maximum TF possible in their corresponding documents.

9. Building posting lists for terms in an inverted index form.

   Structure:
   inverted_index[word] = {doc1: score1, doc2: score2 …}

   The score associated with each document here is: **(g[d] + tf [word, d] * idf [word])/2**

   The scores are normalized by dividing by the maximum score found.

10. The posting lists are sorted and divided into champion and low lists based on the value of ‘**r**’.
11. For every query, the same pre-processing is performed.
12. Value of k is taken as input.
13. A candidate set is formed out of the global champion lists and low lists (if necessary) to find out K contenders for each query term.
14. These individual candidate sets are joined via union and this is the formed set **A**. The final candidates for ranking.
15. Cosine similarity is computed between query terms and the candidates. The top K values are returned.

    TF-IDF value of the query terms are computed as follows:

$$\mathbf{TF = 1/(length\ of\ query)}$$

$$\mathbf{TF\text{-}IDF = TF * IDF(q)}$$

Then for each document, cosine similarity was calculated with the following formula:

$$\textbf{Cosine(doc)} = \textbf{(a1.a1\_ + a2.a2\_+ …)/ } \sqrt{\textbf{(b1.b1 + b2.b2 + …)}} * \sqrt{\textbf{(c1.c1 + c2.c2 + …)}}$$

$$\textbf{Cosine(D, Q) = (D.Q) / (||D||.||Q||)}$$

Where,

a1, a2, a3 … and a1_, a2_, a3_, correspond to the tf-idfs and score values of the terms that are common in the query and the document respectively.
b1, b2, b3 … are final computed scores values of all the document terms and,
c1, c2, c3 … are tf-idf values of all the query terms

Heuristics for value of 'r'

Intuitively, the value of 'r' should be greater than the value of 'k' which is given as input. The results from the champion lists would be empirically better than the results from low lists.

Possible heuristics for r:

- Mean of the dfs of the terms present in vocab.
- Square root of total number of documents.

Pre-Processing

1. Tokenization
2. Lemmatization
3. Punctuation Removal
4. Stop word removal
5. Conversion to lowercase.

Assumptions

1. TF.IDF usage in cosine computation for document terms is switchable with score computed for sorting the lists. This way terms present in high authority docs are given more weightage.
2. Dataset used is **20newsgroups**. All files are used.

Question 2.

Methodology

1. Reading data from the file and processing it.
2. Truncating entries that do not have the qid:4.
3. Storing data of column 75 in a separate list.
4. For total possible files:

   Finding the number of URLS in each score category -> 0, 1, 2, 3 (only these scores were found for qid:4) and multiplying the counts.
5. Printing a sample max DCG file, i.e. the most relevant docs appear first.
6. Computing DCGs and IDCGs – max DCG, for n=50 and for the entire dataset using.
   DCG = r + r1/log2 + r2/log3 + … (log base 2 is used)

7. For computing NDCG, dividing the DCG values by the IDCG values
8. Traversing through entries of column 75 one at a time and computing Precision and Recall values based on the entries seen so far.

   Any document with score greater than 0 is considered relevant and positive.

   Precision = TP / TP+FP
   Recall = TP / TP + FN
9. Plotting the curve

Assumptions

1. Any document with score greater than 0 is considered relevant and positive.