Q1) Champion list

Methodology:

1. Firstly, I have created a dictionary of terms that have a list of document ids whiches hold that term and frequency of that term.
2. Secondly, another dictionary which contains a champion list where documents are sorted based upon Tf of that term in descending order.
3. I have also maintained another list where document names are stored and mapped with their document ids.
4. Now, Both the dictionaries are stored in a pickle file for faster query handling.
5. Another two dictionaries are created for a high list and low list.
6. The high list contains ‘r’ most top-ranked dictionaries for a particular term based upon the highest Tf score.
7. Other documents are stored in a low list.
8. Now, static scores are calculated for those highest ranked documents of each query words and return top ‘k’ documents
9. If the number of retrieved documents are less than ‘k’ then perform step 8 for low list as well and return ‘k’ documents.

Preprocessing Step:

1. Tokenize is done on following delimiters- ("\s", "-", ".", "@", "t", "\n", ‘"’, ">", ",", "?", ":", "{", "(", "[", ")", "}","]", "<", "_", "!", "/", "|", "\", "*", "=", "^")
2. Convert whole text into lower case.
3. Lemmatization is used for both processing query and pre-processing the term.
4. Pickle library is used to store the intermediate inverted index and champion list.

Assumption:

1. I have taken stop words as valid words.
2. To convert query-independent quality score in [0,1], I have divided the number of favorable reviews of a file by maximum number of favorable reviews.
3. To make the champion list, if a doc id is present in high list one of query terms, I have taken that doc into champion list.

Q2) DCG

Methodology:

1. First shortlisted URL related to query id 4 and generate another file.
2. Use DCG formula to calculate DCG for the first 50 URLs and whole URLs.
3. Use DCG formula to calculate DCG of sorted URLs based on the most relevant scores known as the IDCG score.
4. Then Compute nDCG by dividing the DCG score by IDCG score.
5. Then sort URLs of query 4 based on the highest 75th feature and compute precision and recall list.
6. Now I have a computed PR curve based on those lists.

Preprocessing Step:

1. First shortlisted URL related to query id 4 and generate another file.
2. Then fetch 1st and 75th features from URLs to compute precisions and recalls at the different positions.

Assumption:

1. Formula used for computing DCG is rel/log2(pos+1)