INFORMATION RETRIEVAL
Assignment 2 Report

Made By: Pranav Rajput
Roll Number: 19CS30036

Worked on parts : 2A (Implementation and design formulation)
2B (debugging and refactoring)

DETAILS:
(2A)
Design Formulation:
Inverted Index transposition : Space v/s Time ?

I contributed to the design formulation section of the assignment in the beginning where we had to overcome the space complexity of the program. The size of the corpus (set of documents) |S| = 14*10^4 documents and the size of the vocabulary was |T| = 37 * 10^3 words , and if a term-document frequency matrix was constructed for the documents the size of the said matrix would be |S|*|T| = 5.18 * 10^9 elements ~ 38.6 Gb of memory would be allocated to the table, which is not possible to allocate on normal computers. A map was constructed which mapped the cord IDs of the documents with the frequency of the terms in it (in the inverted index itself, with some modifications on the design of the original inverted index as discussed in class, instead of the posting list consisting of <tokens> as elements of the posting list, we insert a tuple <token, frequency>, where the frequency of the token in the document is shipped together with it in the posting list). There was no option but to have a model that compromised on time.

Design Implementation, Problems faced and Solutions

Wrote the code for transposing the inverted index, and grouping the terms with frequencies. Wrote the initial draft for the compute_tf_idf function, which was then further modified by my teammates.

Problem: Running time of the program was very high (90 seconds per iteration) which interfered with our progress as being able to check the status of the variables and the state of the program during runtime was very difficult due to the high running time of the programme.

Solution: Implemented(wrote some of the various methods used by our team to convert numpy vector operations to equivalent dictionary operations) a faster method for the computation of the tf-idf vectors of the documents with dictionaries as we noticed that the amount of operations that were used up in numpy array methods were of a very large order due the vectors being very sparse and these operations were meaningless as we could avoid them altogether with dictionaries. The 90 seconds per iteration was brought down

Debugging:
Debugged the compute_tf_idf function (mismatch of operation type in the computation function were first leading to compilation errors and then to wild values, debugged the index mismatch for the operations in the code)

(2B)
Debugging and Refactoring

The original numpy array for vector realization was switched to the faster dictionary method of computation of the weighing schemes, so I handled the modifications in the code that were required to be changed in order to reduce the run time of the code. I made the changes wherever we used the numpy operations (vectorise, mean, adding vectors) to equivalent operations for dictionaries. I debugged the errors that cropped up after making the said changes (eg.: the type mismatch errors that cropped up when a dict element was treated like a list element and vice versa).