Information Retrieval And Web Search Assignment 1

Jiaming Deng 22302794

1. Introduction

In this assignment, I implement the program to index and search for a given file. I mainly use the lucene framework and the java JDK1.8 and run the program on Microsoft Azure cloud server to get the results.

2. Implement ideas

The first part is to implement the index. First I represent the index content (cran.all.1400) as a class with 5 variables (Id, Title, Author, Bibliography, Words) and read the data in. Then I called the relevant lucene functions to read the data into the lucene document and read the document into the indexwriter.
The second part is to implement the search. First I represent the search content (cran.qry) as a class with 2 variables (Id, queryContent) and read the data in. Finally I call the relevant functions of lucene to search the content and write the results to a file.
Finally, I implement a startup file to run the application, which receives some input args including output file name, analyzer, similarity and update or not and run the functions to solve the problem.
In summary, I divide the code into 5 parts: Model, Parser, Index, Search and a startup file to run the application.

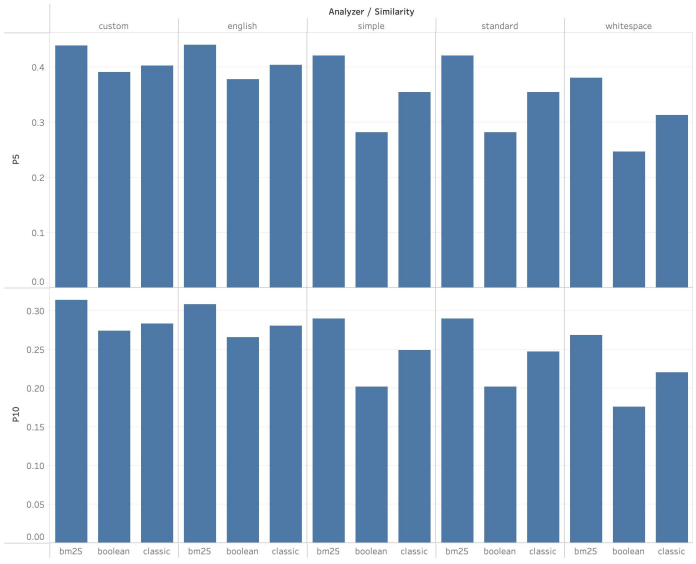
3. Performance of the search engine

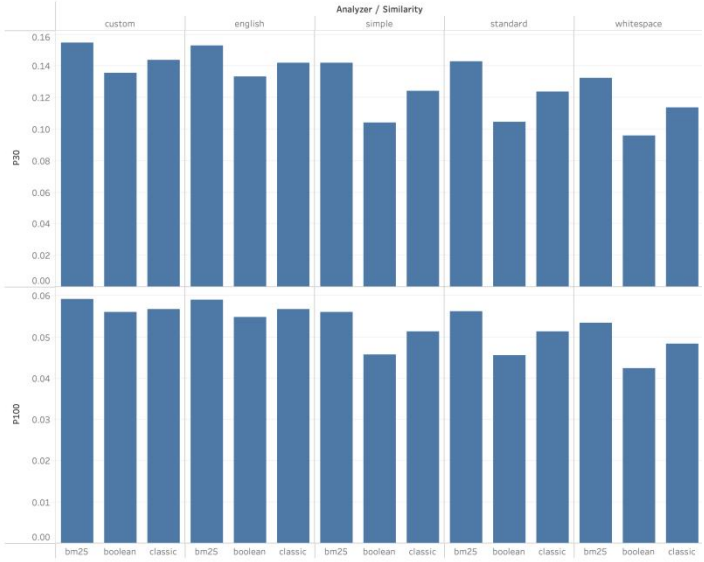
I test the combination of 5 types of analyzers(standard, simple, whitespace, english and custom) and 3 types of similarities(classic, bm25 and boolean) and get 15 results shown in the table below.

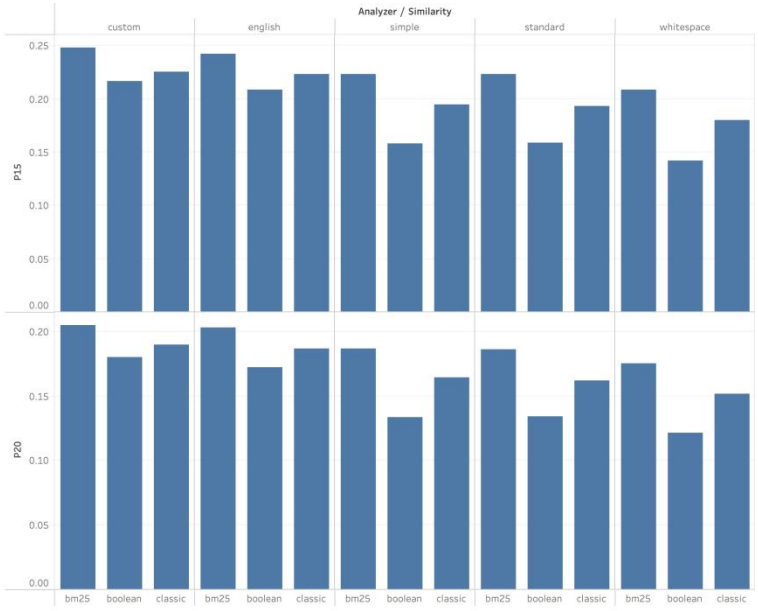
	Analyzer	Similarity	P5	P10	P15	P20	P30	P100
1	standard	bm25	0.4204	0.2893	0.2231	0.1858	0.1427	0.0562
2	standard	classic	0.3547	0.2467	0.1929	0.1618	0.1236	0.0513
3	standard	boolean	0.2818	0.2022	0.1585	0.1338	0.1046	0.0456
4	simple	bm25	0.4204	0.2893	0.2231	0.1864	0.1419	0.0560
5	simple	classic	0.3547	0.2489	0.1944	0.1640	0.1240	0.0513
6	simple	boolean	0.2818	0.2018	0.1579	0.1333	0.1043	0.0457
7	whitespace	bm25	0.3804	0.2680	0.2086	0.1749	0.1324	0.0534
8	whitespace	classic	0.3129	0.2204	0.1799	0.1516	0.1139	0.0484
9	whitespace	boolean	0.2471	0.1760	0.1422	0.1216	0.0961	0.0425
10	english	bm25	0.4400	0.3080	0.2421	0.2027	0.1529	0.0590
11	english	classic	0.4036	0.2800	0.2228	0.1867	0.1422	0.0568
12	english	boolean	0.3778	0.2658	0.2083	0.1722	0.1332	0.0549
13	custom	bm25	0.4382	0.3138	0.2480	0.2047	0.1548	0.0592

14	custom	classic	0.4027	0.2831	0.2252	0.1893	0.1440	0.0568
15	custom	boolean	0.3902	0.2742	0.2166	0.1802	0.1357	0.0560

Precision of the search engines







According the data in the table above, the performance of analyzer is english > custom > standard > simple > whitespace. The reason is english analyzer's TokenStream includes an English Possessive filter, a lower case filter and a stop filter, which is more comprehensive than others'.

The performance of similarity is bm25 > classic > boolean.

4. Conclusion

Overall, according to the experimental data in the table above, the combination of english analyzer and bm25 similarity is the best out of the 15 options.