Sam Pastoriza
Donald Sisco

Information Retrieval Results

We went through several iterations of the search algorithm, utilizing the BM25 scoring function to determine the value of each document. To compute the BM25 scoring function, we needed to determine several key pieces of information, including the number of times that a search term came up in the document. Initially, we decided to split the search query into separate words, count the number of times that each term came up in the document, and add them together to calculate that value. Later, we decided to calculate the skip bigrams given the document, and utilize those to determine the number of times a particular search term came up. If a search query had two or more search terms, we looked for exact matches and weighted those matches much heavier. We also heavily weighted any term found inside of the title tags. Any search term that was found in the title tag was weighted 100,000 times more than if it were found in a different place. When it came to deciding which documents were returned, we rated document by their BM25 scores. We took the maximum score, multiplied it by a confidence interval, and returned all documents that were greater than or equal to that lower bound. We tried confidence intervals between 95% and 99% to see how scoped in our results would become. We ended up with roughly a 98% confidence level, seeing some better results in certain queries.

In the end, the results do show a lot about our search engine. Search terms like “lincoln” came back with documents such as Lincoln and Grant for Lincoln. This makes a lot of sense seeing how intertwined the two individuals were. Some other queries like Taft and Obama returned the singular president as expected. Through the use of bigrams, we got better results with some queries like “emancipation proclamation” and “declaration of independence”. We also encountered a few queries that returned bizarre results. One of the strangest ones was when we searched “patent” we got back Monroe. Due to our lack of any natural language processing, we got the results, “Pierce” and “Polk” when we asked when Abraham Lincoln born.

Example Queries with Results:

Lincoln => [‘Grant.txt’, ‘Lincoln.txt’]
Johnson => ['Grant.txt', 'Johnson.txt', 'Kennedy.txt', 'LBJ.txt']
Bush => ['Bush.txt', 'Carter.txt', 'Clinton.txt', 'GWBush.txt', 'Reagan.txt']
Adams => ['Adams.txt', 'Jefferson.txt', 'JohnQuincyAdams.txt']
Taft => [‘Taft.txt’]
Obama => ['Obama.txt']
Harrison => ['BenjaminHarrison.txt', 'Harrison.txt']
declaration of independence => ['Adams.txt', 'Jefferson.txt']
emancipation proclamation => ['Lincoln.txt']
General => ['Eisenhower.txt', 'Grant.txt', 'Lincoln.txt', 'Washington.txt']

Patent => ['Monroe.txt']
abraham lincoln born => ['Pierce.txt', 'Polk.txt']
War time president => ['Eisenhower.txt', 'Grant.txt']
Vice president => ['Bush.txt', 'Ford.txt', 'GWBush.txt', 'Jefferson.txt', 'Kennedy.txt', 'LBJ.txt', 'Nixon.txt', 'TeddyRoosevelt.txt', 'Truman.txt']
Died in office => ['Jefferson.txt', 'Kennedy.txt']
Assassinated => ['Garfield.txt', 'Harrison.txt', 'McKinley.txt', 'Taft.txt', 'Taylor.txt']
Founding father => ['Adams.txt', 'Madison.txt', 'Monroe.txt']