Nikhil Raina
Information Retrieval
Assessment 2- Retrieval Basics

1. Zipf's distribution, also called Zeta Distribution, is part of the family of the general exponential distributions that are used to model the size of ranks of randomly chosen objects from certain population types. This shows the relative popularity of a small subset of a population. (Glen, 2016)

   Its probability function is shown below:

$$f(x) = \frac{1}{x^{\alpha} \sum_{i=1}^{n} (\frac{1}{i})^{\alpha}}$$

   Where:

   N => is a positive integer

   α => is known as the shape parameter, which is equal to or greater than 0. This determines the shape of the distribution

   The general curve this distribution invokes is as follows:

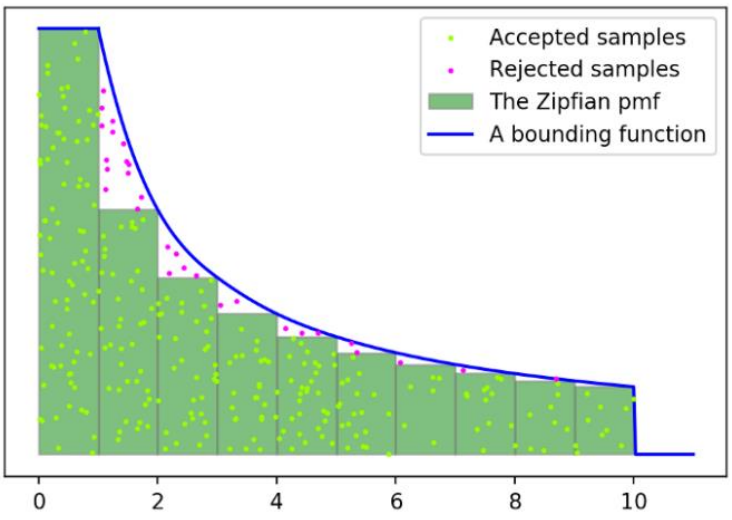

Figure 0.0 (Crease, 2017)

For this assignment, the Figure 0.1 was obtained by plotting frequency of keywords on the x-axis vs the word rank of the corresponding words on the y-axis. Further, the points were arranged in descending order.

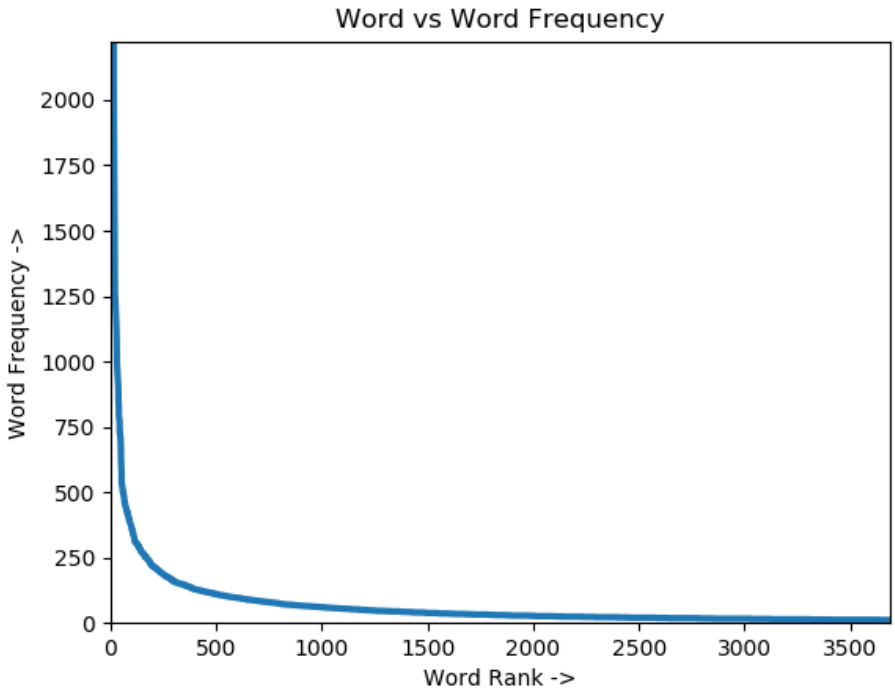

Figure 0.1

By comparing the Figure 0.0 and Figure 0.1, it can be seen that the graph obtained by completing this assessment gives a close resemblance to the Zipr's Distribution. This means that only a few words from the index.tsv file occur very often.

Upon further analysis of the ordering of the frequency of the words, it was found that number such as 4, 5 and 6 had the highest frequency. Ideally, according to the definition, a stop word is a commonly used word that a search engine has been programmed to ignore, both when indexing entries for searching and while retrieving them as the result of a search query (Dutta, 2020). However, setting these number as stop words would be a bad idea as they could give crucial value for the search query, such as dates, magnitude to a unit, justifying a quantity, etc. Instead of these, it would be better to put articles such as 'a', 'of' and 'the' as stopping words as they were the next most frequent words to be counted during the indexing procedure.

In this assessment, the processing of the text could be handled better. There should be a few pre-decided stopping words that could have been removed while indexing. To further better the processing and understanding of indexing, lemmatization could be used. This would allow words to drop down to their more basic or dictionary form. However, this requires extra computational linguistics power that can identify the part of speech. (Heidenreich, 2018)

2. The following are the results for Disjunctive queries (OR):
    a. With ‘retrieval’ as the query

```
Execution Time:  0.22340154647827148
Total unique matched documents:  131

1. 238 Inefficiency of the Use of Boolean Functions 1
2. 274 Dynamic Storage Allocation for an Information Retrieval System 1
3. 291 An Information Retrieval Language for Legal Studies 1
4. 439 Record Linkage 1
5. 618 Retrieval of Misspelled Names in an Airlines Passenger Record System 3
6. 625 A Method of Representation, Storage and Retrieval 1
7. 633 Manipulation of Trees in Information Retrieval* 1
8. 650 A Survey of Languages and Systems for Information Retrieval 1
9. 652 Translation of Retrieval Requests Couched 1
10. 654 COMIT as an IR Language 1
11. 674 Coding Clinical Laboratory Data For Automatic Storage and Retrieval 1
12. 797 A Catalogue Entry Retrieval System 1
13. 890 Everyman's Information Retrieval System 5
14. 891 RECOL-A Retrieval Command Language 2
15. 943 Storage and Search Properties of a Tree-Organized Memory System 1
16. 1031 Theoretical Considerations in Information Retrieval Systems 3
17. 1167 Across Machine Lines in COBOL 1
18. 1193 Establishment of the ACM Repository and Principles 1
19. 1235 The SMART Automatic Document Retrieval System-An Illustration 3
20. 1270 Secondary Key Retrieval Using an IBM 7090-1301 System 2
```

Figure 1.0

    b. With ‘information retrieval’ as the query

```
Execution Time:  0.22440028190612793
Total unique matched documents:  319

1. 207 An Introduction to Information Processing Language V 1
2. 238 Inefficiency of the Use of Boolean Functions 2
3. 271 A Storage Allocation Scheme for ALGOL 60 1
4. 274 Dynamic Storage Allocation for an Information Retrieval System 2
5. 291 An Information Retrieval Language for Legal Studies 2
6. 395 Automation of Program  Debugging 1
7. 396 A Card Format for Reference Files in Information Processing 3
8. 408 CL-1, An Environment for a Compiler 1
9. 439 Record Linkage 3
10. 615 An Information Algebra - Phase I Report-Language 2
11. 633 Manipulation of Trees in Information Retrieval* 2
12. 650 A Survey of Languages and Systems for Information Retrieval 2
13. 651 Use of Semantic Structure in Information Systems 1
14. 654 COMIT as an IR Language 2
15. 655 An Information System With The Ability To Extract Intelligence From Data 1
16. 656 Information Structures for Processing and Retrieving 1
17. 669 Some Legal Implications of the Use of Computers in the Banking Business 1
18. 674 Coding Clinical Laboratory Data For Automatic Storage and Retrieval 2
19. 689 USA Participation in an International 1
20. 695 An Automatic Data Acquisition and Inquiry System Using Disk Files 2
```

Figure 1.1

    c. With ‘information retrieval retrieval’ as the query

```
Execution Time:  0.22690176963806152
Total unique matched documents:  319

1. 207 An Introduction to Information Processing Language V 1
2. 238 Inefficiency of the Use of Boolean Functions 3
3. 271 A Storage Allocation Scheme for ALGOL 60 1
4. 274 Dynamic Storage Allocation for an Information Retrieval System 3
5. 291 An Information Retrieval Language for Legal Studies 3
6. 395 Automation of Program  Debugging 1
7. 396 A Card Format for Reference Files in Information Processing 3
8. 408 CL-1, An Environment for a Compiler 1
9. 439 Record Linkage 4
10. 615 An Information Algebra - Phase I Report-Language 2
11. 633 Manipulation of Trees in Information Retrieval* 3
12. 650 A Survey of Languages and Systems for Information Retrieval 3
13. 651 Use of Semantic Structure in Information Systems 1
14. 654 COMIT as an IR Language 3
15. 655 An Information System With The Ability To Extract Intelligence From Data 1
16. 656 Information Structures for Processing and Retrieving 1
17. 669 Some Legal Implications of the Use of Computers in the Banking Business 1
18. 674 Coding Clinical Laboratory Data For Automatic Storage and Retrieval 3
19. 689 USA Participation in an International 1
20. 695 An Automatic Data Acquisition and Inquiry System Using Disk Files 2
```

Figure 1.2

d. With ‘information retrieval compression’ as the query

```
Execution Time:  0.22539758682250977
Total unique matched documents:  328

1. 207 An Introduction to Information Processing Language V 1
2. 238 Inefficiency of the Use of Boolean Functions 2
3. 271 A Storage Allocation Scheme for ALGOL 60 1
4. 274 Dynamic Storage Allocation for an Information Retrieval System 2
5. 291 An Information Retrieval Language for Legal Studies 2
6. 395 Automation of Program  Debugging 1
7. 396 A Card Format for Reference Files in Information Processing 3
8. 408 CL-1, An Environment for a Compiler 1
9. 439 Record Linkage 3
10. 615 An Information Algebra - Phase I Report-Language 2
11. 633 Manipulation of Trees in Information Retrieval* 2
12. 650 A Survey of Languages and Systems for Information Retrieval 2
13. 651 Use of Semantic Structure in Information Systems 1
14. 654 COMIT as an IR Language 2
15. 655 An Information System With The Ability To Extract Intelligence From Data 1
16. 656 Information Structures for Processing and Retrieving 1
17. 669 Some Legal Implications of the Use of Computers in the Banking Business 1
18. 674 Coding Clinical Laboratory Data For Automatic Storage and Retrieval 2
19. 689 USA Participation in an International 1
20. 695 An Automatic Data Acquisition and Inquiry System Using Disk Files 2
```

Figure 1.3

3. The following are the results for conjunctive queries (AND):
    a. With ‘retrieval’ as the query

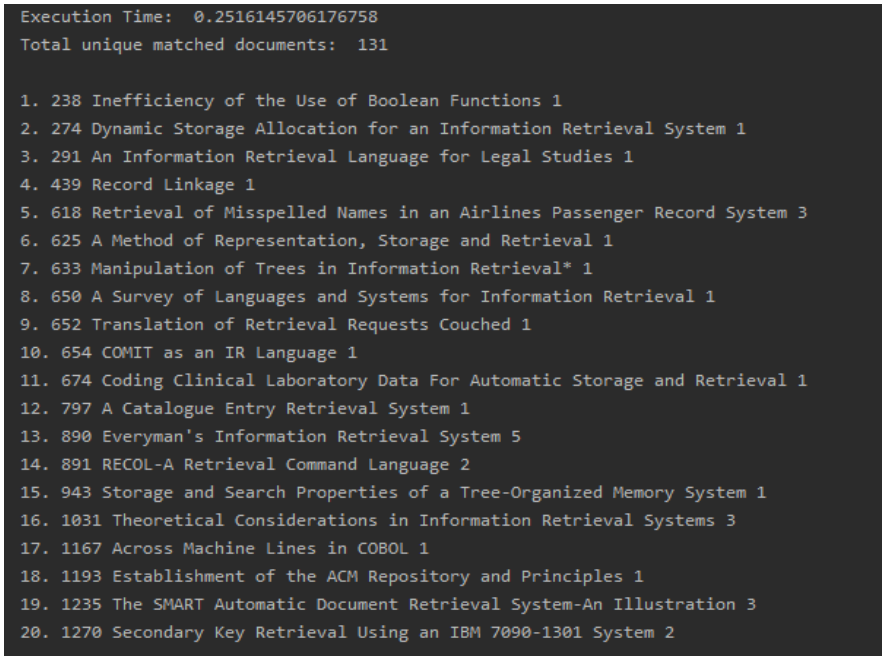

Figure 2.0

b. With ‘information retrieval’ as the query

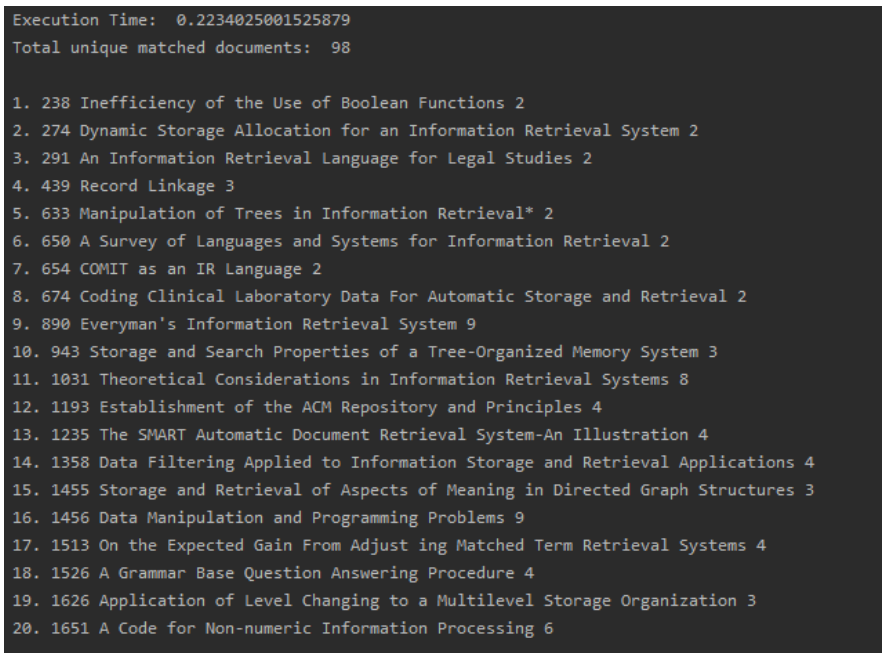

Figure 2.1

c. With ‘information retrieval retrieval’ as the query

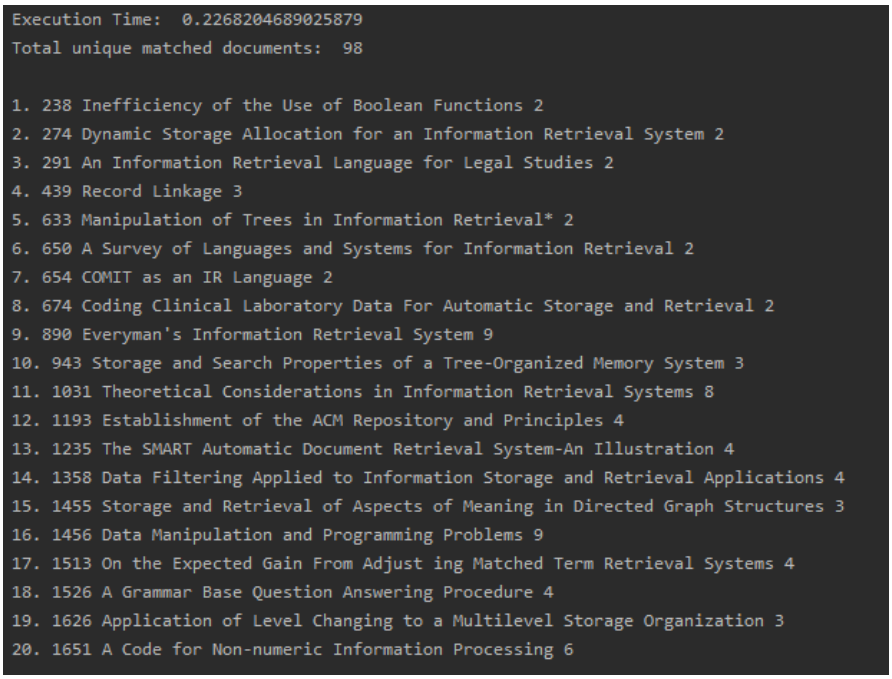

Figure 2.2

d. With 'information retrieval compression' as the query

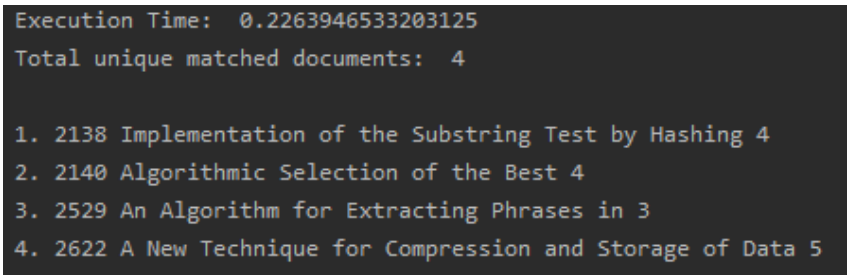

Figure 2.3

All the above results, separately, show the Execution time, number of unique matched documents, the Document ID, title of the document, score assigned to the document and the ranking of the document from the top 20 search results of that specific query.

From the results above, it was observed that Figure 1.0 and Figure 2.0 got the same amount of uniquely matched documents where Figure 1.0's Disjunctive Query search was 0.03 seconds faster. Not only that, but the relative scores of each document is the same. This makes sense as there was only one query to search for. This behavior isn't consistent with the other searches as the length of the query increases and bring in similar word search as well.

The second query of 'Information Retrieval' obtain different results when comparing the output of the Disjunctive and Conjunctive searches. Figure 1.1 returns 319 matches whereas Figure 2.1 returns only 98. There are a few similarities between the top 20 results, especially the top 5. This kind of behavior, with regards to the difference of uniquely match documents was as expected.

The third query minted a similar result as query two's did. A few visible differences are in the execution time where the third query taken 0.04 seconds longer to deliver the result. The order of the top 20 matches turned out to be slightly different, where the documents with the higher frequency for the keyword 'retrieval' were ranked higher.

For the final query, a massive difference was observed. The figure 1.3 was able to collect 328 uniquely matched documents whereas the Figure 2.3 was able to only collect 4. This meant that the keyword 'compression' must have been rarely used in the same documentation as the keywords 'information' and 'retrieval'.

The overall behavior and results that were obtained from the respective searches were went according to plan. The reduction in the matches of keywords in the conjunctive search was as perceived as it was searching the entire set of keywords that occurred in a document, unlike the disjunctive search method where it searched of each keyword as is and didn't look at the entire set as one unit for search.

Work Cited:

Crease, J. (2017, July 8). *Rejection-sampling The Zipf Distribution*. Medium. https://medium.com/@jasoncrease/rejection-sampling-the-zipf-distribution-6b359792cffa

Dutta, A. (2020, May 18). *Removing Stop Words With NLTK In Python - GeeksforGeeks*. GeeksforGeeks. https://www.geeksforgeeks.org/removing-stop-words-nltk-python/#:~:text=What%20are%20Stop%20words%3F,result%20of%20a%20search%20query.

Gallagher, J. (2020, July 28). *How To Sort A Dictionary By Value In Python | Career Karma*. Career Karma. https://careerkarma.com/blog/python-sort-a-dictionary-by-value/

Glen, S. (2016, July 14). *Zeta Distribution (Zipf Distribution) - Statistics How To*. StatisticsHowTo.Com. https://www.statisticshowto.com/zeta-distribution-zipf/

Heidenreich, H. (2018, December 21). *Stemming? Lemmatization? What?* Towards Data Science. https://towardsdatascience.com/stemming-lemmatization-what-ba782b7c0bd8

kgkmeekg. (2019, July 23). *Python Code To Remove HTML Tags From A String*. Stack Overflow. https://stackoverflow.com/questions/9662346/python-code-to-remove-html-tags-from-a-string