4NL3 Assignment 1

Guneev Arora - 400477579

January 20, 2025

Contents

1 Data **2**

2 Methodology **2**
2.1 Tokenization And Normalization 2
2.2 Reading The File And Arguments 3
2.3 Figures 3

3 Sample Output **4**
3.1 Top and Bottom 25 Words 4
3.2 Figures 5

4 Discussion **7**
4.1 Findings 7
4.2 What I Learned 8

1 Data

For my file, I used "The Project Gutenberg eBook of Modern Billiards" from as The Project Gutenberg as mentioned. This book is on the game billiards. I chose this book as it is a game I enjoy playing and has a lot of knowledge on the game and its history.
The book can be found at https://www.gutenberg.org/ebooks/59143 for public use.
This book also has a total of 84223 tokens according to my tokenization that I will discuss in the later content of the report.

2 Methodology

In this section I Discuss the approach I took to writing my software and generating my figures. What are the options that the user can select when performing the text normalization steps.

2.1 Tokenization And Normalization

To begin with, I started with tokenization which I did with a simple python ".split" command to my data.

Now in terms of the normalization tools, I created these five normalization tools that can be used by the users:

1. Lowercase: Makes all the text lowercase.
2. Stemming: Gets rid of the suffixes of the words in the data.
3. Lemmatize Words: Lemmatizes the words in the data.
4. Remove Stop words: Removes common words that do not mean much such as "and".
5. Remove Numbers And Punctuation and single letters: Removes all the separate punctuation and numbers from the data along with single letters.

As we were told to implement the first four tools, I added those using the NLTK library along with basic python function ".lower" for lowercase.

These used a very basic approach with libraries and a simple "for" in python and also used some content from lecture three to implement.

For the last tool I chose to implement Remove Numbers And Punctuation and single letters, as I wanted to remove all when working on the assignment as I thought they were useful, from trial and error along with what the data from tested had such as duplicate words, random numbers and also random single letters, so I implemented this tool using basic python tools "isdigit" and "string.punctuation".

2.2 Reading The File And Arguments

Firstly to read the file, I began with a python ".open" command with "r" meaning to read and to be able to access its data I also used encoding="utf-8". Now, to be able to take user arguments and name of the file I used the python library "sys" to take in the arguments. If a valid file name is not inputted or a file that doesn't exist, It raises an error that tells the user the problem.

Now in terms, of the arguments I took the arguments spoken about in the previous subsection. These can be input using `--<argument/normalization name>` and writing `python <normalize.py> <bookname.txt>` and then any of the five normalization tools you want using the above format `--<argument/normalization name>`.

To ensure that there no errors, I used a few else cases that raise an error if a user doesn't input valid or correct arguments/normalization tool names. And to store the names I used a dictionary to store the names along with True or False values can be used later for the normalization.

2.3 Figures

For my figures, I created two figures both using the matplotlib library.
I created one with the bar plot where the x-axis is the rank of the word in your list and the y-axis is the frequency and log.
The second figure has the frequency of the top 25 words in the data.
For getting the data I stored it in a Counter from the Collections library as it said allowed to be used according to comments in lecture three.

Other than that, it also send a message in the terminal with every word along with

its frequency which was again done using the Collections library.

3 Sample Output

First I will give you my output for the first/top 25 words and last/bottom 25 words in the data with all filters and also no filters.

3.1 Top and Bottom 25 Words

Top 25 Words - no filters/arguments:

the: 3270	*: 2525	and: 2136	of: 1979
in: 1443	to: 1415	a: 1108	at: 706
by: 611	for: 592	is: 510	was: 460
on: 449	with: 409	The: 393	as: 354
A.: 353	be: 320	from: 313	ball: 294
or: 279	W.: 279	object-ball: 270	N.: 266
it: 262			

Bottom 25 Words - no filters/arguments:

donations.: 1	donate," 1	please: 1	visit:: 1
Professor: 1	originator: 1	library: 1	shared: 1
anyone.: 1	loose: 1	network: 1	volunteer: 1
editions,: 1	confirmed: 1	included.: 1	edition.: 1
PG: 1	facility:: 1	Gutenberg™,: 1	eBooks,: 1
subscribe: 1	email: 1	newsletter: 1	hear: 1
eBooks.: 1			

Top 25 Words - with all filters/arguments:

game: 520 first: 353 championship: 342 thi: 318 side: 275 may: 218 right: 199	wa: 460 averag: 353 tournament: 341 hall: 313 run: 241 left: 215	ball: 453 vs: 352 carom: 338 play: 309 match: 232 city: 210	stroke: 388 objectbal: 350 cushion: 326 cuebal: 285 one: 231 second: 206

Bottom 25 Words - with all filters/arguments:

commit: 1 compliance: 1 swamp: 1 checks: 1 loo: 1 edition: 1 hear: 1	chariti: 1 unsolicit: 1 web: 1 card: 1 network: 1 pg: 1	charit: 1 grate: 1 current: 1 donations: 1 editions: 1 facility: 1	paperwork: 1 treatment: 1 addresses: 1 anyone: 1 included: 1 newslett: 1

3.2 Figures

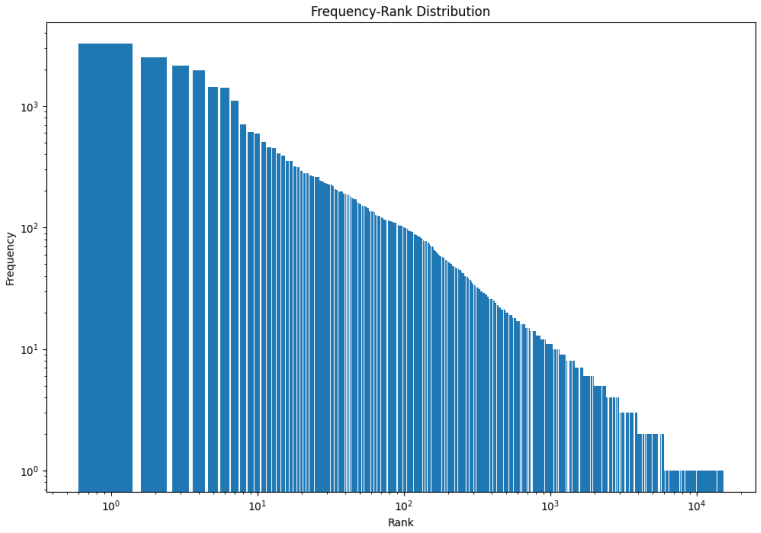

Figure 1: Bar Plot of frequency and rank when there are no filters/arguments

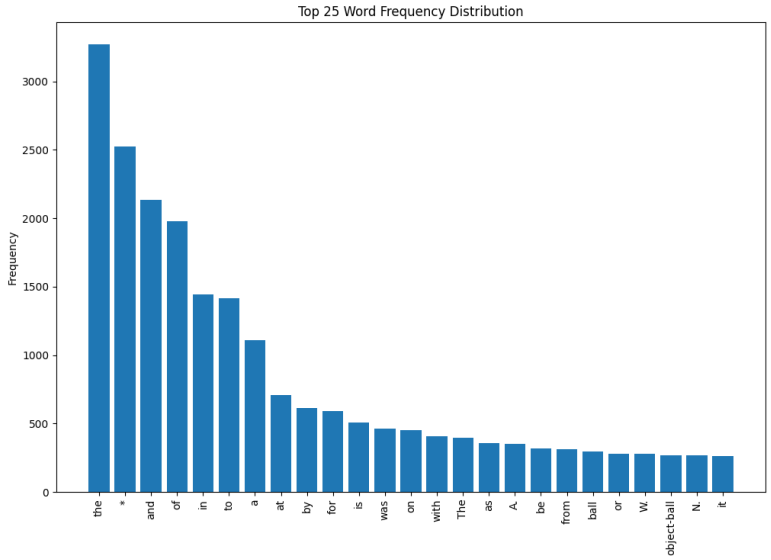

Figure 2: Top 25 Words in the data when there are no filters/arguments

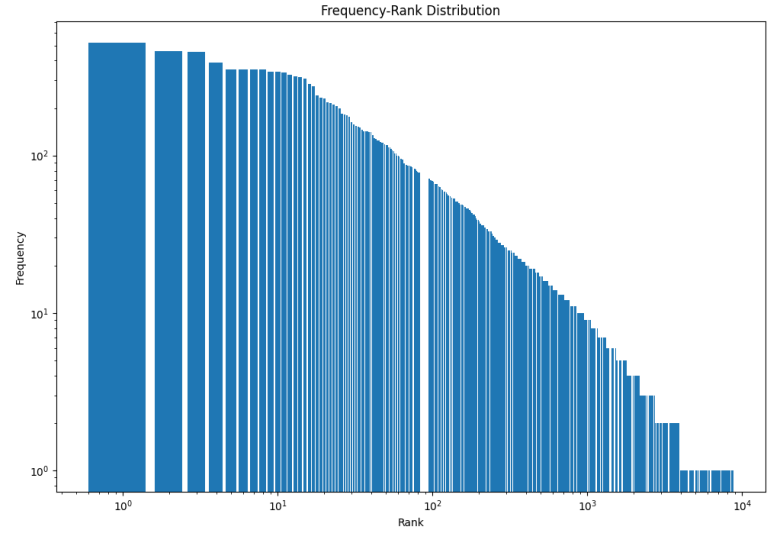

Figure 3: Bar Plot of frequency and rank when there are no filters/arguments

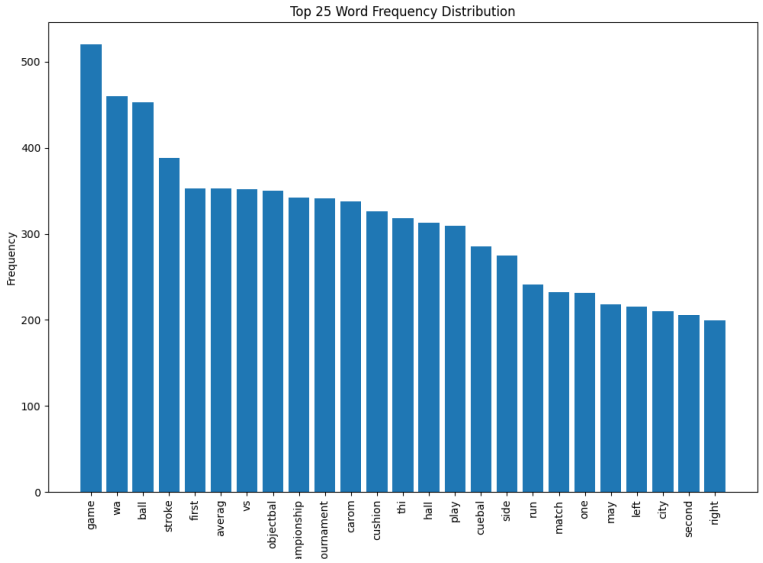

Figure 4: Top 25 Words in the data when there are no filters/arguments

4 Discussion

4.1 Findings

From all the collected sample output, without any filters, most the common/top words make sense, they are mostly generic. However, there are some exceptions such as "*", "A.", "W." which are not common and also not useful in the data without any meaning in my opinion as it is hard to get any context from them which is why I got rid of punctuations and single letters together as when removing punctuations turned them into single letters. Now for the other side of the table, they were mostly random words that just weren't used much or had some duplicate words with symbols or punctuations that weren't duplicated. When using all filters this just left random words that were not used much by the end of it, with some errors caused by the filters too.

Now, comparing my data to the Wikipedia data, I have found that there are similarities from the "Word frequencies in natural languages" section. Mainly the shape of my graph compared to theirs for the log frequency against the log rank, following the Zipf distribution for both no filters and also all filters. This is due to as mentioned on

the Wikipedia "the most common word occurs about n times the n-th most common one." When using all the filters, the graph starts with a slower decent as the rank decreases, as a lot of common words are removed.

4.2 What I Learned

From the assignment, I learned about a lot of topics such as how to do tokenization, and also how to normalize data using python using basic python functions and also the NLTK library to begin with. Other than that I also learned how to properly use terminal arguments for the first time as I hadn't done that before in python.

I also learned a lot about the Zipf distribution and law and on how it works and is used which was really enjoyable for me to learn about. All of this was really cool to learn and also implement in practice. I had a few struggles, some of them were in the data and the reason I decided to add the filter to normalize that I wanted to, and the other which didn't think would be one was finding a book with over 50,000 tokens which was a bit funny in my opinion, but I got it after a few tries. Overall, I had a lot of fun working on this assignment and putting topics into practice.

For AI tools, I used ChatGPT, I did around 32 queries which each are 4.26g = 136.32 grams of CO2 total with all of them being queries on some coding questions but not directly copying code and some other questions on certain topics.