SFWRENG 4NL3 Assignment 1

Sumanya Gulati

20 January 2025

Contents

List of Tables

List of Figures

1 Data

`myfile.txt` is a text file that contains inaugral speeches by United States Presidents' from George Washington to Barack Obama. The dataset was acquired from the Project Gutenberg collection and was downloaded as a Plain Text UTF-8 encoded file.

Prior to tokenization, the dataset was manually pre-processed which included removing the title page, table of contents, disclaimers and such information from the beginning and the ending of the text file to ensure consistency and remove any characters unrelated to the actual content.

I chose this dataset in light of today's inaugral speech of the 47th U.S. President. I also wanted to be able to use the script from this assignment to gather insights about how inaugral addresses have changed over centuries and analyze whether the increased use of words like 'immigration' and 'economy' is a recent trend over the past few decades.

I also want to analyze if the use of words propogating sentiments of nationalism and unity increases in years following major national of global events like the Vietnam war, 9/11, the 2008 recession, COVID-19 pandemic and more. If you would be interested in viewing the results of these analyses, feel free to shoot me an email and I would be glad to share them (assuming, I am actually able to get it done soon).

2 Methodology

This section outlines the approach employed in writing this script and explains available normalization options while describing their use and a rationale for using them.

2.1 Approach

The approach I used to write this script involved breaking it down into steps and implementing one required step at a time, tetsing it and then moving onto the next step. I started by writing a skeletal script that reads a file and implements command-line argument parsing. After that, I wrote the `tokenize_text()` function to pattern-match the input text file and break it into tokens, storing the result as a list.

The next step involved implementing the *lowercase* and *remove_stopwords* options, followed by implementing the *lemmatization* and *stemming* options, the code snippets for which were copied from the lecture slides. The `visualize()` function was then implemented to produce first, a DataFrame (basically, a nicer table to view the output) and then a bar plot. A trial and error approach was used to figure out if either or both axes should be set as a log scale and a log-log line plot was generated to compare my result with that of datasets obeying Zipf's law as shown in this Wikipedia article.

Generative AI was used to resolve a logical bug and write line 12 of the script. A search query weas employed using ChatGPT with the following prompt: *"I am trying to tokenize an input text file using regex pattern matching but some of my tokens in the list are preceeded by the letters, ufeff. Could this be random or is there a logical error in my code?"*. As outlined in the course outline, the carbon footprint of this query is 4.32g of CO_2.

2.2 Available Options

The following options, in any combination, can be used by the user during the text normalization steps:

1. `--lowercase`: Change letter case
2. `--stem`: Apply stemming
3. `--lemmatize`: Apply lemmatization

4. `--remove-stopwords`: Remove commonly used stopwords in the English language

5. `--remove-numbers`: Remove numbers

Beyond the defined requirements, I added the option to remove numbers. Since the text file contains speeches spanning centuries, it seemed like a fair assumption (backed by a brief look at the raw data) that the text file contains the use of digits to write out for example, the date or the year of the speech. I believe this is a useful option as it removes the numbers added as a part of the description, producing a more consistent dataset that focuses purely on the content of the speeches by exluding the remaining descriptive text.

I recognize that this option may cause unwanted issues by removing numbers that are a part of the actual speech or not remove numbers that are spelled out as opposed to being written ad digits. Configuring this option to not cause these issues seems like a more complex task that would require additional time and effort.

3 Sample Output

Using the `pandas` library and the DataFrame structure, a table has been created to sort the list of unique tokens in decreasing order along with their frequency/count.

Figure 1 displays a table showing the top 25 words and their frequency based on the program's output.

	Tokens	Count	Rank
0	the	9396	1
1	of	7047	2
2	and	5091	3
3	to	4384	4
4	a	3078	5
5	in	2553	6
6	our	1977	7
7	that	1743	8
8	it	1580	9
9	be	1473	10
10	is	1434	11
11	we	1191	12
12	for	1101	13
13	by	1051	14
14	have	1004	15
15	which	1003	16
16	not	940	17
17	with	898	18
18	will	869	19
19	I	834	20
20	are	795	21
21	all	773	22
22	their	726	23
23	this	713	24
24	The	623	25

Figure 1: Top 25 words with their count.

Similarly, 2 displays a table showing the last 25 words and their frequency based on the program's output.

	Tokens	Count	Rank
8821	Falls	1	8822
8822	Selma	1	8823
8823	Stonewall	1	8824
8824	sung	1	8825
8825	unsung	1	8826
8826	footprint	1	8827
8827	preacher	1	8828
8828	King	1	8829
8829	inextricably	1	8830
8830	daughter	1	8831
8831	gay	1	8832
8832	student	1	8833
8833	workforce	1	8834
8834	expelled	1	8835
8835	Detroit	1	8836
8836	Appalachia	1	8837
8837	lane	1	8838
8838	Newtown	1	8839
8839	contour	1	8840
8840	absolutism	1	8841
8841	reasoned	1	8842
8842	Philadelphia	1	8843
8843	realizes	1	8844
8844	END	1	8845
8845	P	1	8846

Figure 2: Last 25 words with their count.

Using the `matplotlib.pyploy` library, a bar graph depicted in figure 3 shows the top 45 words. It must be noted that the y-axis has been set to a log scale to better understand the generated output. Although a normal bar plot without a log scale did produce a readable graph, the difference between frequencies was not as apparent.

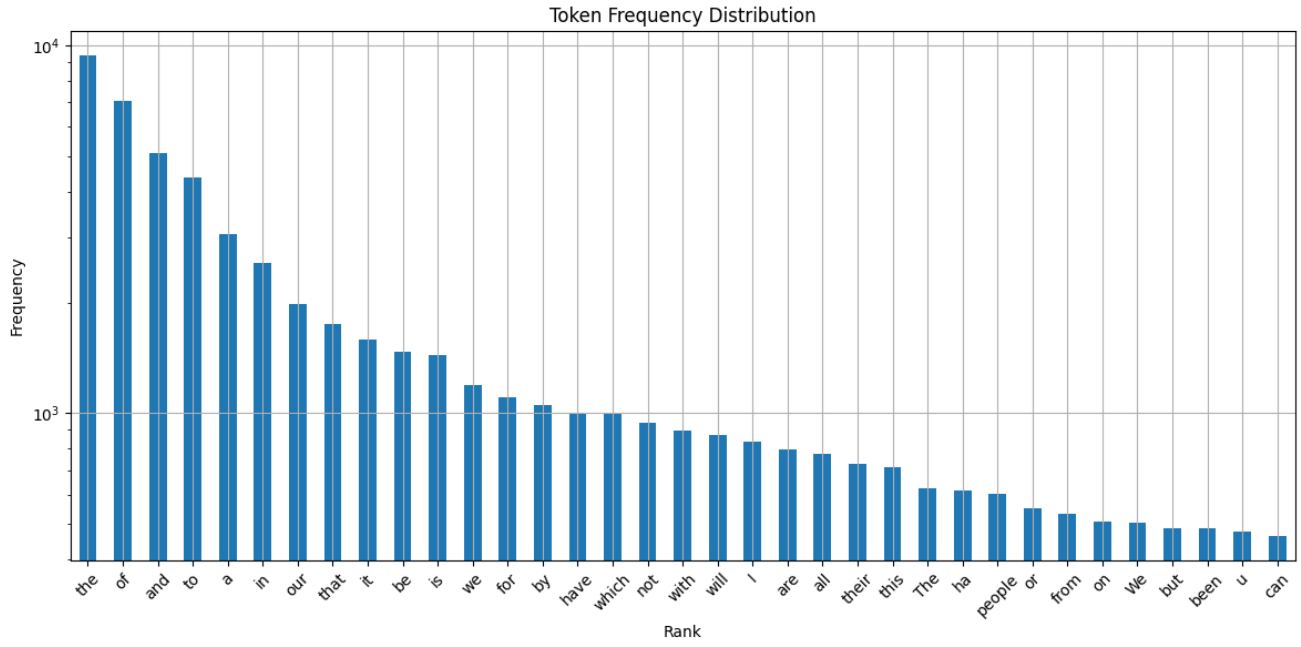

Figure 3: Bar plot showing the frequencies of top 45 words.

To better compare the output with Zipf's Law, a line graph as shown in figure 4 has been generated. It must be noted that this is a log-log plot.

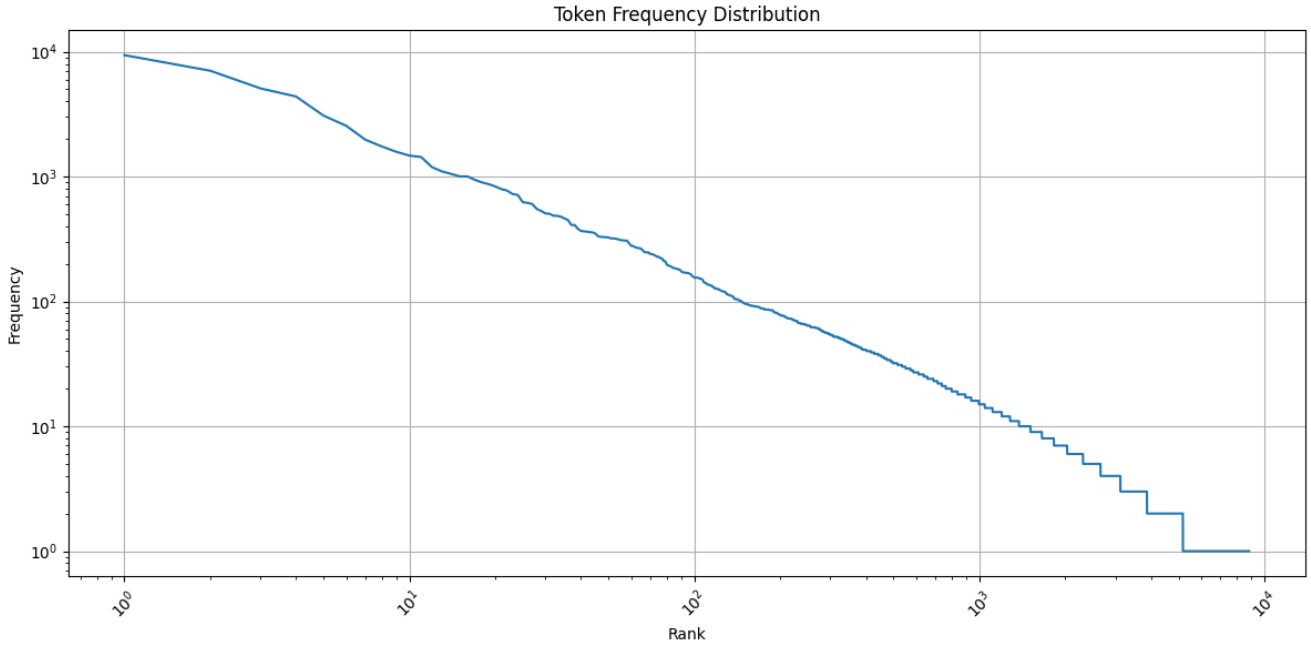

Figure 4: Last 25 words with their count.

4 Discussion

4.1 Findings

4.2 Reflection