

BIS 420 PROGRAMMING FOR DATA SCIENCE
PRAJAKTA POHARE
CHAPTER 13 EXERCISE 13.9
ILLINOIS STATE UNIVERSITY

The “rank” of a word is its position in a list of words sorted by frequency: the most common word has rank 1, the second most common has rank 2, etc. Zipf’s law describes a relationship between the ranks and frequencies of words in natural languages (http://en.wikipedia.org/wiki/Zipf's_law). Specifically, it predicts that the frequency, f , of the word with rank r is:

$f = cr^{-s}$ where s and c are parameters that depend on the language and the text. If you take the logarithm of both sides of this equation, you get:

$\log f = \log c - s \log r$. So if you plot $\log f$ versus $\log r$, you should get a straight line with slope $-s$ and intercept $\log c$.

Write a program that reads a text from a file, counts word frequencies, and prints one line for each word, in descending order of frequency, with $\log f$ and $\log r$. Use the graphing program of your choice to plot the results and check whether they form a straight line. Can you estimate the value of s ?

```
import string

import math

import matplotlib.pyplot as plt

def load_book(book):

    with open('/Users/prajaktapohare/Library/CloudStorage/OneDrive-ILStateUniversity/BIS420/Week 13/book.txt', 'r', encoding='utf-8') as f:

        lines = f.readlines()

        start = 0

        end = len(lines)

        for i, line in enumerate(lines):

            if "*** START OF" in line:

                start = i + 1
```

```

elif "*** END OF" in line:

    end = i

    break

return ".join(lines[start:end])

def clean_text(text):

    translator = str.maketrans(" ", string.punctuation)

    return text.translate(translator).lower()

def count_words(text):

    words = text.split()

    word_count = {}

    for word in words:

        word_count[word] = word_count.get(word, 0) + 1

    return word_count

def compute_zipf(word_counts):

    sorted_words = sorted(word_counts.items(), key=lambda x: x[1], reverse=True)

    log_ranks = []

    log_freqs = []

    print(f'{"Rank":<6} {"Word":<15} {"Freq":<10} {"log(R)":<10} {"log(F)":<10}')

    print("-" * 50)

    for rank, (word, freq) in enumerate(sorted_words, start=1):

        log_r = math.log(rank)

        log_f = math.log(freq)

        log_ranks.append(log_r)

        log_freqs.append(log_f)

```

```

    if rank <= 20:

        print(f' {rank:<6} {word:<15} {freq:<10} {log_r:<10.4f} {log_f:<10.4f} ')

    return log_ranks, log_freqs


def plot_zipf(log_ranks, log_freqs, book_title="Zipf's Law"):

    plt.figure(figsize=(8,6))

    plt.plot(log_ranks, log_freqs, marker='o', linestyle='-', color='blue')

    plt.title(f'Zipf's Law Plot: {book_title}')

    plt.xlabel("log(Rank)")

    plt.ylabel("log(Frequency)")

    plt.grid(True)

    plt.tight_layout()

    plt.show()


book_file = '/Users/prajaktapohare/Library/CloudStorage/OneDrive-ILStateUniversity/BIS420/Week 13/book.txt'

raw = load_book(book_file)

cleaned = clean_text(raw)

word_counts = count_words(cleaned)


log_r, log_f = compute_zipf(word_counts)

plot_zipf(log_r, log_f, book_title=book_file)

```

```

Users > prajaktapohare > Library > CloudStorage > OneDrive-ILStateUniversity > BIS420 > Week 13 > BIS420_PrajaktaPohare_Ch13_13.9.py > clean_text
1 import string
2 import math
3 import matplotlib.pyplot as plt
4
5 def load_book(book):
6     with open('/Users/prajaktapohare/Library/CloudStorage/OneDrive-ILStateUniversity/BIS420/Week 13/book.txt', 'r', encoding='utf-8') as f:
7         lines = f.readlines()
8         start = 0
9         end = len(lines)
10        for i, line in enumerate(lines):
11            if "*** START OF" in line:
12                start = i + 1
13            elif "*** END OF" in line:
14                end = i
15                break
16        return ''.join(lines[start:end])
17
18 def clean_text(text):
19     translator = str.maketrans('', '', string.punctuation)
20     return text.translate(translator).lower()
21
22 def count_words(text):
23     words = text.split()
24     word_count = {}
25     for word in words:
26         word_count[word] = word_count.get(word, 0) + 1
27     return word_count
28
29 def compute_zipf(word_counts):
30     sorted_words = sorted(word_counts.items(), key=lambda x: x[1], reverse=True)
31     log_ranks = []
32     log_freqs = []
33     print(f'{"Rank":<6}{"Word":<15}{"Freq":<10}{"log(R)":<10}{"log(F)":<10}{"log(F)/log(R)":<10}')
34     print("-" * 50)
35     for rank, (word, freq) in enumerate(sorted_words, start=1):

```

```

Users > prajaktapohare > Library > CloudStorage > OneDrive-ILStateUniversity > BIS420 > Week 13 > BIS420_PrajaktaPohare_Ch13_13.9.py > clean_text
29 def compute_zipf(word_counts):
30     print(f'{"Rank":<6}{"Word":<15}{"Freq":<10}{"log(R)":<10}{"log(F)":<10}{"log(F)/log(R)":<10}')
31     print("-" * 50)
32     for rank, (word, freq) in enumerate(sorted_words, start=1):
33         log_r = math.log(rank)
34         log_f = math.log(freq)
35         log_ranks.append(log_r)
36         log_freqs.append(log_f)
37         if rank <= 20:
38             print(f'{"rank":<6}{"word":<15}{"freq":<10}{"log_r":<10.4f}{"log_f":<10.4f}{"log_f/log_r":<10.4f}')
39     return log_ranks, log_freqs
40
41 def plot_zipf(log_ranks, log_freqs, book_title="Zipf's Law"):
42     plt.figure(figsize=(8,6))
43     plt.plot(log_ranks, log_freqs, marker='o', linestyle='-', color='blue')
44     plt.title(f'Zipf's Law Plot: {book_title}')
45     plt.xlabel("log(Rank)")
46     plt.ylabel("log(Frequency)")
47     plt.grid(True)
48     plt.tight_layout()
49     plt.show()
50
51 book_file = '/Users/prajaktapohare/Library/CloudStorage/OneDrive-ILStateUniversity/BIS420/Week 13/book.txt'
52 raw = load_book(book_file)
53 cleaned = clean_text(raw)
54 word_counts = count_words(cleaned)
55
56 log_r, log_f = compute_zipf(word_counts)
57 plot_zipf(log_r, log_f, book_title=book_file)
58

```