Assignment 2

**Answer 1:**

According to the dictionary, Plagiarism is practice of taking someone else’s work or ideas and passing them off as one’s own. In the given scenario, we need to verify if the passages in the given document are either duplicates or near duplicates of any other document in the document database. Here, we are comparing document passages instead of the whole document against any document.

Basically, we need to find if there is any passage in a given document which matches any of the passages of all documents provided in the document collection. If we can find any passage in any other document containing same passage for which the probability of being duplicate is close to one, we can define it as plagiarized passage.

Utilizing the slightly modified version of fingerprinting method, we can find the probability. As described in the lectures.

1. Given a document we should parse the document into indexed passages and then parsed words for all words in each of the passage. It needs to remove non-word content, punctuation, HTML tags and spaces.

2. Each of the words is grouped into contiguous n-grams for some n grams. Some n grams may be overlapping.

3. Some of the n-grams are selected to represent the passage.

4. The selected n-grams are hashed to improve retrieval efficiency and further can reduce the size of the paragraph.

5. These hash values are stored as an index for each passage of document combined with its passage index.

6. Using overlap of fingers, the document is compared against all other document in the document store.

**Answer 2 a**.

Please refer to attached file named “ZipLaw.py” for the code. And “AssignOutput.txt” files for the output of the assignment.

Total number of unique of words found: 2632

Total number of words found in the dictionary: 26693

According to Zipf’s law, Rank (r) of words times its frequency (k) is constant.

In the text Alice in Wonderland, Considering the rank of word “the”

For word “the”, Rank\*frequency = 1\*1644 = 1644

For word “and”, Rank\* frequency = 2\*872 =1644

**Answer 2 b.**

According to Zipf’s law, Number of words that occur n times has rank Rn = k/n

Number of words with frequency n is = Rn – Rn+1 ----- (1)

= k/n – k/n+1

= k/n (n+1)

Calculating on the basis of zip’f law,

Pr = probability of occurrence

Tt = total number of words

Rank\*Pr = C

Since Pr = frequency/total number of words

Putting value of Pr in the formula we get,

Rank\*frequency = C\*Tt

k = C\*Tt ----- (2)

From (1)

Number of words with frequency 1 = k/n (n+1)

= (C\*Tt)/2

Number of words with frequency 2 = k/n (n+1)

= (C\*Tt)/6

Number of words with frequency 3 = k/n (n+1)

= (C\*Tt)/12

Number of words with frequency 4 = k/n (n+1)

= (C\*Tt)/20

Adding above four equations, we get

Total number of words with frequency less than 5 = (C\*Tt)\*(1/2+1/6+1/12+1/20)

= (C\*Tt)\*0.8

Assuming C = 0.1

The proportion of words omitted = (number of words with frequency less than 5)/Total number of words

The Proportion of words in percent = (0.1\*0.8\*Tt)\*100/Tt

= 81%

Now, calculating the proportion of words omitted from “alice in wonderland” text

Again using the formula, number of words = Rn – Rn+1

Number of words with frequency 1 = 2632 -1487 = 1145 \* 1 = 1145

Number of words with frequency 2 = 1484 – 1079 = 405\*2 = 810

Number of words with frequency 3 = 1079 – 846 = 233\*3 = 699

Number of words with frequency 4 = 846 – 697 = 149\*4 = 596

Tf, Total number of words with frequency less than five = 1145+810+699+596 = 3250

Proportion of words omitted in Percent = (Number of words with frequency less than five)/ total number of words

= (3250/26693)\*100

= 12.15%

**Answer 3a:**

According to Heap’s law, v = k \* n^b, Here

v = vocubalary size i.e number of unique words

n = number of words in corpus

k and beta are the parameters that vary for each corpus.

Using Heap’s law,

v = k \* n^0.5, where b = 0.5

When we encounter 90% of vocabulary

0.90 v = k \* n1^0.5

The proportion of words need to be read will be

(0.90 \* v)/v = (k\*n1^0.5)/ (k\*n^0.5)

0.90 = (n1) ^0.5/ (n) ^0.5

Taking square root on both side

0.81 \* n = n1

Hence atleast 81 % of text should be read to encounter 90% of the vocab.

**Answer 4a:**

I used Bing search engine to estimate the result size for the below queries.

We know that we can estimate the size of the result set using the formula,

fabc =  (fab \* fbc)/fb

1. “Music Cake Strings”

Result size for documents given by Bing = 3,180,000

fmusic = 1,080,000,000

fcake = 134,000,000

fstring = 46,600,000

fmusic \_cake = 52,700,000

fcake\_strings = 6,540,000

fmusic\_strings = 52,900,000

Substituting in the above formula,

fmusic\_cake\_strings  = ( fmusic\_cake  \* fcake\_strings)/ fcake

= (52,700,000 \* 6,540,000)/134,000,000

= 2,572,074

Comparing the calculated and actual result we found that since the words were almost independent, so the actual and calculated result is near.

2. “Mobile Application Development”

Result size for documents given by Google = 260,000,000

fmobile = 6,310,000,000

fapplication = 1,880,000,000

fdevelopment = 2,120,000,000

fmobile\_application = 820,000,000

fapplication\_development = 577,000,000

fmobile\_development = 5,640,000,000

Substituting in the above formula,

fmobile\_application\_development = ( fmobile\_application  \* fapplication\_development)/ fapplication

= (820,000,000\*577,000,000)/1,880,000,000

= 251,670,000

Comparing the calculated and actual result we found results given by google search engine for the dependent words is near to the estimated result size.

**Answer 4b:**

Since we know to calculate the collection size, we use formula

N2 = (fa \* fb \* fc)/ fabc

Substituting the values for First Query, i.e. Music Cake Strings

N2 = (1,080,000,000 \* 134,000,000 \* 46,600,000)/3,180,000

N = 460,513,843

Substituting the values for Second Query, i.e. Mobile Application Development

N2 = (63,100,000,000 \* 1,880,000,000 \* 2,120,000,000)/3,180,000

N = 31101035352.54