CMSC 676
HOMEWORK 1

Submitted By:-
Prajna Bhandary
QC21457

1. SUMMARY

The program is written in python using the jupyter notebook. The main aim of the assignment is to tokenize the words of different files and find out the frequency of the words used in all the files. The program accepts two parameters in the command line namely the address to the input directory and the output directory. I have used html2text for tokenization. The efficiency of the program is calculated with respect to the time taken by the program to execute a set number of files. The final results including the distributed tokens and the distributed frequency are stored in two seperate files. The output file also includes a directory of all tokenized documents.

1.1 Handling punctuations and numbers

First, the text was extracted from the files using html2text. Second, using the list of all possible punctuation and numbers stated, all the unwanted characters were replaced with a whitespace wherever they occured. Next, the tokens were split based on whitespace. All words are also converted to lower case to avoid the redundancy of words based on the cases. Some characters considered are:
escape_char =
["\n","@","\t","\r",",",".","/","-","_","'",'"',"`","[","]","(",")","?",":","|","*",";","!","{","}",">","$","=","%","#","+","<","&","\\","0","1","2","3","4","5","6","7","8","9"]

1.2 Calculating frequency of each word

The frequency of each word was calculated using a hash map. Every new token is added on to the list with their value and the frequency of the number of times it arrives. If a word has come across for the first time then the word is added to the list with the value 1. With every successive word that is already in the list, the frequency count is incremented.

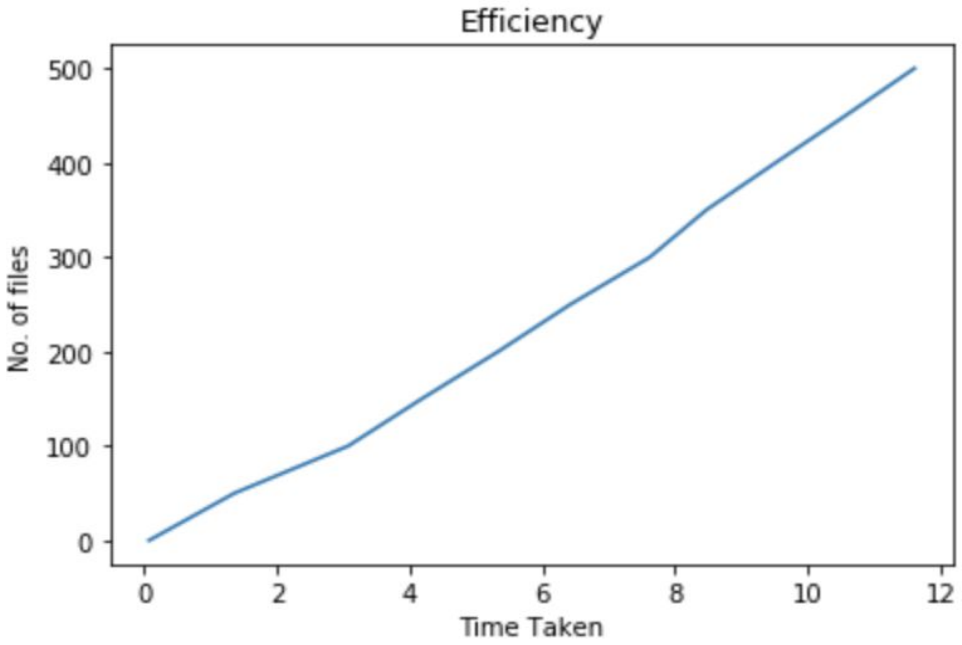
1.3 Incorrect Tokenization

There are some tokens with random tokens such as ‘aaaeealktqti’ that occurs 20 times or ‘elem’ that occurs 5 times and so on. This could be due to the reason that the the escape characters are replaced with a whitespace and the split for tokens are happening at a whitespace.

1.4 Efficiency of the program

To calculate the efficiency of the program I have considered the time taken to process every 50 number of files. The graph showing the results can be seen as shown below:

```
Text(0.5, 1.0, 'Efficiency')
```



As we can observe in the above graph the efficiency of the program seems to be linear to any size file.

The distributed tokens and distributed frequencies are added in their respective files. The distributed tokens are saved in ascending order and distributed frequencies are saved in descending order according to the highest frequencies at the top of the list.

2. Results

The following observations are made in my program:

- total time taken by my program to execute is approx. 12 sec.
- The word ‘the’ had the highest frequency with the number of occurrences as 33143. Followed by a, of, and, com and so on.

2.1. Comparison

I compared my results with Sai Madhurya. She used the programming language as python and used Beautiful Soup as the parser. The following observations were made in her program:

- The time taken by her program was approx. 23 secs.
- The highest frequency word is ‘the’ with the frequency 33531.

In comparison, my program runs faster than her program. This could be because she used beautiful soup whereas I used html2text. It could also be because of the programming style. The frequency of the word ‘the’ was the highest in both our cases but she got a higher number of the word. Again it could be a reason with the parsing tool used.

3. Conclusion

Even though html2text was an efficient tokenizer, beautiful soup probably gave a better result with the tokenization.