**HOMEWORK 1 TEXT MINING WITH PYTHON**

**By- Zeeshan Ali**

[alisz@mail.uc.edu](mailto:alisz@mail.uc.edu)

18. Using list addition, and the set and sorted operations, compute the vocabulary of the sentences sent1 ... sent8.

**Program:**

#Adding list of sent

sentmy=sent1+sent2+sent3+sent4+sent5+sent6+sent7+sent8

#unique and sorted

sentmy2=sorted(set(sentmy))

#Counting Vocabulary

len(sentmy2)

**Output:**

**sentmy=sent1+sent2+sent3+sent4+sent5+sent6+sent7+sent8**

**#unique and sorted**

**sentmy2=sorted(set(sentmy))**

**#Counting Vocabulary**

**len(sentmy2)**

**Out[26]: 75**

19. What is the difference between the following two lines? Which one will give a larger value? Will this be the case for other texts?

**Program:**

#converting all words in lower from text1, getting distinct words and sorting them

len(sorted(set(w.lower() for w in text1)))

#Getting distinct words, changing them in lower case and then sorting them

len(sorted(w.lower() for w in set(text1)))

#This will have more value because some letters for e.g. This and this will not be considered same in set #function operation and they will be changed to lower case later hence they will be retained

**#**This will be case for other texts as well

**Output:**

**len(sorted(set(w.lower() for w in text1)))**

**Out[29]: 17231**

**len(sorted(w.lower() for w in set(text1)))**

**Out[30]: 19317**

**20. What is the difference between the following two tests: w.isupper() and not w.islower()?**

**Program:**

#This will check whether it is uppercase or not

"Zeeshan".isupper()

#This will just invert the result of islower function

not "Zeeshan".islower()

**Output:**

**"Zeeshan".isupper()**

**Out[41]: False**

**"Zeeshan".islower()**

**Out[42]: False**

**not "Zeeshan".islower()**

**Out[43]: True**

21. Write the slice expression that extracts the last two words of text2

**Program:**

#This will extract last two words of text2

text2[-2:]

**Output:**

**text2[-2:]**

**Out[45]: ['THE', 'END']**

**22.** Find all the four-letter words in the Chat Corpus (text5). With the help of a frequency distribution (FreqDist), show these words in decreasing order of frequency

**Program:**

#Getting all four letter words

z=[w.lower() for w in text5 if len(w) == 4]

#Frequency Distribution of text5 words

z1=FreqDist(z)

#Sorting based on values from high to low

sorted(z1.items(), key=lambda x:x[1], reverse=True)

**Output:**

**Only displaying portion of output, the key value pair will have the frequency of word from high to low**

**('heys', 1),**

**("<3's", 1),**

**('lisa', 1),**

**('brwn', 1),**

**('hurr', 1)]**

24. Write expressions for finding all words in text6 that meet the conditions listed below. The result should be in the form of a list of words: ['word1', 'word2', ...].

1. Ending in *ize*
2. Containing the letter *z*
3. Containing the sequence of letters *pt*
4. Having all lowercase letters except for an initial capital (i.e., titlecase)

**Program:**

#Ending with 'ize'

sorted(w for w in set(text6) if w.endswith('ize'))

#Containing 'z'

sorted(term for term in set(text6) if 'z' in term)

#Containing 'pt'

sorted(term for term in set(text6) if 'pt' in term)

#Containing first letter capital other letters as lowercase

sorted(item for item in set(text6) if item.istitle())

**Output:**

**sorted(w for w in set(text6) if w.endswith('ize'))**

**Out[97]: []**

**sorted(term for term in set(text6) if 'z' in term)**

**Out[98]: ['Fetchez', 'amazes', 'frozen', 'zhiv', 'zone', 'zoo', 'zoop', 'zoosh']**

**sorted(term for term in set(text6) if 'pt' in term)**

**Out[99]:**

**['Chapter',**

**'Thpppppt',**

**'Thppppt',**

**'Thpppt',**

**'Thppt',**

**'aptly',**

**'empty',**

**'excepting',**

**'ptoo',**

**'temptation',**

**'temptress']**

**Displaying portion of output as it is quite long**

**Out[100]:**

**['A',**

**'Aaaaaaaaah',**

**'Aaaaaaaah',**

**'Aaaaaah',**

**'Aaaah',**

**'You',**

**'Your',**

**'Yup',**

**'Zoot']**

25.  Define sent to be the list of words ['she', 'sells', 'sea', 'shells', 'by', 'the', 'sea', 'shore']. Now write code to perform the following tasks:

1. Print all words beginning with *sh*
2. Print all words longer than four characters

**Program:**

#Defining list sent

sent=['she', 'sells', 'sea', 'shells', 'by', 'the', 'sea', 'shore']

#Starting with 'sh'

[w for w in set(sent) if w.startswith('sh')]

#If word longer than four letters

[w for w in set(sent) if len(w)>4]

**Output:**

**[w for w in set(sent) if w.startswith('sh')]**

**Out[103]: ['shells', 'shore', 'she']**

**[w for w in set(sent) if len(w)>4]**

**Out[104]: ['shells', 'sells', 'shore']**

26. What does the following Python code do? sum(len(w) for w in text1) Can you use it to work out the average word length of a text?

**Program:**

#It is summing up the length of all words of text1

sum(len(w) for w in text1)

#Average length of word in text1

sum(len(w) for w in text1)/len(text1)

**Output:**

**sum(len(w) for w in text1)**

**Out[111]: 999044**

**sum(len(w) for w in text1)/len(text1)**

**Out[112]: 3.830411128023649**

**The average length of a word is 3.83**

27. Define a function called vocab\_size(text) that has a single parameter for the text, and which returns the vocabulary size of the text.

**Program:**

#Defining function vocab\_size

def vocab\_size(text):

return len(set([word.lower() for word in text]))

#Calling function

vocab\_size(text1)

**Output:**

**vocab\_size(text1)**

**Out[121]: 17231**

28.  Define a function percent(word, text) that calculates how often a given word occurs in a text, and expresses the result as a percentage.

**Program:**

#Function to calculate word %

def percent(word, text):

return 100 \* text.count(word) / len(text)

#CAlling function

print(str(percent("the",text1))+'%')

**Output:**

**print(str(percent("the",text1))+'%')**

**5.260736372733581%**