

NLP LS-1 UNDERSTANDING LARGE TEXTFILES

Roll No : 225229111

Exercise-1

```
In [1]: import nltk
```

```
In [7]: nltk.download('wordnet')
text="this is Andrew's text,isn't it?"
```

```
[nltk_data] Downloading package wordnet to
[nltk_data] C:\Users\1mscdsa46\AppData\Roaming\nltk_data...
[nltk_data] Package wordnet is already up-to-date!
```

```
In [9]: #1.
tokenizer=nltk.tokenize.WhitespaceTokenizer()
tokens=tokenizer.tokenize(text)
print(len(tokens))
print(tokens)

5
['this', 'is', "Andrew's", "text,isn't", 'it?']
```

```
In [21]: #2.
tokenizer=nltk.tokenize.TreebankWordTokenizer()
tokens=tokenizer.tokenize(text)
print(tokens)

['this', 'is', 'Andrew', "'s", 'text', ',', 'is', "n't", 'it', '?']
```

```
In [20]: #3.
tokenizer=nltk.tokenize.WordPunctTokenizer()
tokens=tokenizer.tokenize(text)
print(tokens)

['this', 'is', 'Andrew', "'", 's', 'text', ',', 'isn', "'", 't', 'it', '?']
```

Exercise-2

```
In [22]: f=open("gift-of-magi.txt","r")
con=f.read()
print(con)
```

The Gift of the Magi
by O. Henry

One dollar and eighty-seven cents. That was all. And sixty cents of it was in pennies. Pennies saved one and two at a time by bulldozing the grocer and the vegetable man and the butcher until one's cheeks burned with the silent imputation of parsimony that such close dealing implied. Three times Della counted it. One dollar and eighty-seven cents. And the next day would be Christmas.

There was clearly nothing left to do but flop down on the shabby little couch and howl. So Della did it. Which instigates the moral reflection that life is made up of sobs, sniffles, and smiles, with sniffles predominating.

While the mistress of the home is gradually subsiding from the first stage to the second, take a look at the home. A furnished flat at \$8 per week. It did not exactly beggar description, but it certainly had that word on the lookout for the mendicancy squad.

In the vestibule below was a letter-box into which no letter would go, and an

```
In [23]: #2.
#1)
tokenizer=nltk.tokenize.WhitespaceTokenizer()
tokens=tokenizer.tokenize(con)
print(len(tokens))
```

2074

```
In [30]: #2)
from nltk import *
data=FreqDist(tokens)
print(data)
```

<FreqDist with 956 samples and 2074 outcomes>

```
In [26]: #3)
data.most_common(20)
```

```
Out[26]: [('the', 107),
('and', 74),
('a', 64),
('of', 51),
('to', 41),
('was', 26),
('she', 25),
('in', 24),
('had', 21),
('her', 21),
('that', 20),
('it', 19),
('at', 19),
('with', 19),
('for', 19),
('his', 17),
('on', 16),
('I', 14),
('Jim', 13),
('were', 11)]
```

```
In [28]: #4)
from nltk import *
test=[w for w in tokens if len(w)>10]
freq=FreqDist(test)
freq
```

```
Out[28]: FreqDist({'Dillingham': 2,
                  '"Sofronie."': 1,
                  'Christmas!': 1,
                  'appertaining': 1,
                  'brilliantly.': 1,
                  'calculated.': 1,
                  'close-lying': 1,
                  'contracting': 1,
                  'critically.': 1,
                  'description': 1,
                  'description.': 1,
                  'difference?': 1,
                  'disapproval.': 1,
                  'duplication.': 1,
                  'eighty-seven': 3,
                  'grandfather's.': 1,
                  'illuminated': 1,
                  'inconsequential': 1,
                  'intoxication': 1,
                  'laboriously.': 1,
                  'longitudinal': 1,
                  'mathematician': 1,
                  'men--wonderfully': 1,
                  'meretricious': 1,
                  'necessitating': 1,
                  'ornamentation--as': 1,
                  'possession.': 1,
                  'possessions': 1,
                  'predominating.': 1,
                  'proclaiming': 1,
                  'sterling--something': 1,
                  'tortoise-shell.': 1,
                  'twenty-two--and': 1,
                  'wonderfully': 1})
```

```
In [29]: #5)
for i,j in freq.items():
    if len(i) > 10 and j>2:
        print(i,j)
```

eighty-seven 3

Exercise-3

STEP-1

```
In [34]: fname ="austen-emma.txt"
         f=open(fname, 'r')
         etxt=f.read()
         print(etxt)
         f.close()
```

[Emma by Jane Austen 1816]

VOLUME I

CHAPTER I

Emma Woodhouse, handsome, clever, and rich, with a comfortable home and happy disposition, seemed to unite some of the best blessings of existence; and had lived nearly twenty-one years in the world with very little to distress or vex her.

She was the youngest of the two daughters of a most affectionate, indulgent father; and had, in consequence of her sister's marriage, been mistress of his house from a very early period. Her mother had died too long ago for her to have more than an indistinct remembrance of her caresses; and her place had been supplied by an excellent woman as governess, who had fallen little short of a mother in affection.

```
In [36]: etxt[-200:]
```

```
Out[36]: 'e deficiencies, the wishes,\nthe hopes, the confidence, the predictions of the\nsmall band\nof true friends who witnessed the ceremony, were fully answered\nin\nthe perfect happiness of the union.\n\n\nFINIS\n'
```

```
In [43]: nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]   C:\Users\1mscdsa46\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt.zip.
```

```
Out[43]: True
```

```
In [44]: etoks=nlk.word_tokenize(etxt.lower())  
etoks[-20:]
```

```
Out[44]: ['of',  
          'true',  
          'friends',  
          'who',  
          'witnessed',  
          'the',  
          'ceremony',  
          ',',  
          'were',  
          'fully',  
          'answered',  
          'in',  
          'the',  
          'perfect',  
          'happiness',  
          'of',  
          'the',  
          'union',  
          '.',  
          'finis']
```

```
In [46]: len(etoks)
```

```
Out[46]: 191669
```

```
In [49]: etypes=sorted(set(etoks))  
etypes[-10:]
```

```
Out[49]: ['younger',  
          'youngest',  
          'your',  
          'yours',  
          'yourself',  
          'yourself.',  
          'youth',  
          'youthful',  
          'zeal',  
          'zigzags']
```

```
In [50]: len(etypes)
```

```
Out[50]: 8000
```

```
In [51]: efreq=nlk.FreqDist(etoks)  
efreq['beautiful']
```

```
Out[51]: 24
```

STEP-2

Question 1: words with prefix and suffix

```
In [52]: [word for word in etoks if word.startswith("un") & word.endswith("able")]
```

```
Out[52]: ['unexceptionable',  
          'unsuitable',  
          'unreasonable',  
          'unreasonable',  
          'uncomfortable',  
          'unfavourable',  
          'unexceptionable',  
          'unexceptionable',  
          'uncomfortable',  
          'unpersuadable',  
          'unavoidable',  
          'unreasonable',  
          'uncomfortable',  
          'unsuitable',  
          'unmanageable',  
          'unexceptionable',  
          'unreasonable',  
          'unobjectionable',  
          'unpersuadable',  
          'unsuitable',  
          'unreasonable',  
          'uncomfortable',  
          'unexceptionable',  
          'unpardonable',  
          'unmanageable',  
          'unanswerable',  
          'unfavourable',  
          'unpersuadable',  
          'unaccountable',  
          'undesirable',  
          'unable',  
          'unable',  
          'unpardonable',  
          'unexceptionable',  
          'unreasonable',  
          'unreasonable',  
          'uncomfortable',  
          'unreasonable',  
          'unpardonable',  
          'unaccountable',  
          'unexceptionable',  
          'unreasonable',  
          'unaccountable']
```

Question 2: Length

```
In [53]: tokenizer=nltk.tokenize.WordPunctTokenizer()  
toke=tokenizer.tokenize(etxt)
```

```
In [54]: [word for word in token if len(word)>15]
```

```
Out[54]: ['companionableness',
          'misunderstanding',
          'incomprehensible',
          'undistinguishing',
          'unceremoniousness',
          'Disingenuousness',
          'disagreeableness',
          'misunderstandings',
          'misunderstandings',
          'misunderstandings',
          'misunderstandings',
          'disinterestedness',
          'unseasonableness']
```

Question 3: Average word length

```
In [55]: average=sum(len(word)for word in token)/len(token)
          average
```

```
Out[55]: 3.755268231589122
```

Question 4: Word frequency

```
In [56]: from nltk import *
          fdieem=FreqDist(token)
```

```
In [57]: for i,j in fdieem.items():
          if j>200:
              print(i,j)
```

```
Emma 865
by 558
Jane 301
I 3178
Woodhouse 313
, 11454
and 4672
with 1187
a 3004
to 5183
some 248
of 4279
the 4844
; 2199
had 1606
- 574
one 413
in 2118
very 1151
1111 254
```



```
In [66]: e2gramfd.most_common(20)
```

```
Out[66]: [(',', 'and'), 1879),
          (('Mr', '.'), 1153),
          ('"', 's'), 932),
          (';', 'and'), 866),
          ('.', '"'), 757),
          (('Mrs', '.'), 699),
          (('to', 'be'), 595),
          ('.', 'I'), 570),
          (',', 'I'), 568),
          (('of', 'the'), 556),
          (('in', 'the'), 434),
          (';', 'but'), 427),
          ('.', 'Weston'), 426),
          ('.', '"'), 415),
          ('.', 'She'), 413),
          (',', '"'), 398),
          (('I', 'am'), 395),
          ('.', 'Elton'), 381),
          (',', 'that'), 359),
          (('had', 'been'), 308)]
```

Question 8: Bigram frequency count

```
In [67]: for i,j in e2gramfd.items():
          if i==('so','happy'):
              print(i,j)
```

```
('so', 'happy') 4
```

Question 9: Word following 'so'

```
In [73]: import re
          from collections import Counter
```

```
In [75]: words=re.findall(r'so+ \w+',open('austen-emma.txt').read())
          ab=Counter(zip(words))
          print(ab)
```

```
Counter({'so m',): 138, ('so v',): 78, ('so l',): 61, ('so s',): 61, ('so
f',): 49, ('so w',): 43, ('so a',): 40, ('so p',): 36, ('so t',): 31, ('so
e',): 31, ('so d',): 31, ('so g',): 31, ('so o',): 29, ('so c',): 29, ('so
i',): 29, ('so h',): 25, ('so k',): 15, ('so n',): 15, ('so r',): 14, ('so
I',): 14, ('so b',): 13, ('so u',): 7, ('so y',): 7, ('so j',): 2, ('so _',):
2, ('so q',): 2, ('so P',): 1})
```

Question 10: Trigrams

```
In [79]: e3grams=list(nltk.trigrams(tokens))
e3gramfd=nltk.FreqDist(e3grams)
```

```
In [81]: last_ten=FreqDist(dict(e3gramfd.most_common()[-10:]))
last_ten
```

```
Out[81]: FreqDist({'answered', 'in', 'the'): 1,
                  ('ceremony,', 'were', 'fully'): 1,
                  ('fully', 'answered', 'in'): 1,
                  ('in', 'the', 'perfect'): 1,
                  ('of', 'the', 'union.'): 1,
                  ('perfect', 'happiness', 'of'): 1,
                  ('the', 'ceremony,', 'were'): 1,
                  ('the', 'perfect', 'happiness'): 1,
                  ('the', 'union.', 'FINIS'): 1,
                  ('were', 'fully', 'answered'): 1})
```

Question 11: Trigram top frequency

```
In [82]: e3gramfd.most_common(10)
```

```
Out[82]: [ (('I', 'do', 'not'), 94),
           (('I', 'am', 'sure'), 75),
           (('would', 'have', 'been'), 55),
           (('a', 'great', 'deal'), 55),
           (('she', 'could', 'not'), 49),
           (('could', 'not', 'be'), 45),
           (('she', 'had', 'been'), 44),
           (('it', 'would', 'be'), 43),
           (('do', 'not', 'know'), 43),
           (('Mr.', 'and', 'Mrs.'), 37)]
```

Question 12: Trigram frequency count

```
In [83]: words1 = re.findall(r'so happy to \w+', open('austen-emma.txt').read())
print(words1)
```

```
[]
```