Name : RETHINAGIRI G

Roll no : 225229130

Course Title : Natural Language pre-processing Lab

Lab : 01. Understanding Large Text Files

exercise 1

In [1]:

```
import nltk
```

In [2]:

```
nltk.download('wordnet')
```

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

Out[2]:

```
True
```

In [3]:

```
word="This is Andrew's text, isn't it ?"
word
```

Out[3]:

```
"This is Andrew's text, isn't it ?"
```

In [4]:

```
Tokenizer=nltk.tokenize.WhitespaceTokenizer()
tokens=Tokenizer.tokenize(word)
print(len(tokens))
print(tokens)
```

```
7
['This', 'is', "Andrew's", 'text,', "isn't", 'it', '?']
```

In [5]:

```
Tokenizer=nltk.tokenize.TreebankWordTokenizer()
tokens=Tokenizer.tokenize(word)
print(len(tokens))
```

```
10
```

In [6]:

```
Tokenizer=nltk.tokenize.WordPunctTokenizer()
tokens=Tokenizer.tokenize(word)
print(len(tokens))
```

```
12
```

Exercise-2

In [7]:

```
file=open("C:\\Users\\user\\Downloads\\gift-of-magi.txt","r")
m=file.read()
m
```

Out[7]:

```
'The Gift of the Magi\nby O. Henry\n\nOne dollar and eighty-seven cents. That was all. And sixty cents of it was in penni
es. Pennies saved one and two at a time by bulldozing the grocer and the vegetable man and the butcher until one\'s cheek
s burned with the silent imputation of parsimony that such close dealing implied. Three times Della counted it. One dolla
r and eighty-seven cents. And the next day would be Christmas.\n\nThere was clearly nothing left to do but flop down on t
he shabby little couch and howl. So Della did it. Which instigates the moral reflection that life is made up of sobs, sni
ffles, and smiles, with sniffles predominating.\n\nWhile the mistress of the home is gradually subsiding from the first s
tage 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 look-out for the mendicancy squad.\n\nIn the vestibule below was a letter-box into whic
h no letter would go, and an electric button from which no mortal finger could coax a ring. Also appertaining thereunto w
as a card bearing the name "Mr. James Dillingham Young."\n\nThe "Dillingham" had been flung to the breeze during a former
period of prosperity when its possessor was being paid $30 per week. Now, when the income was shrunk to $20, the letters
of "Dillingham" looked blurred, as though they were thinking seriously of contracting to a modest and unassuming D. But w
henever Mr. James Dillingham Young came home and reached his flat above he was called "Jim" and greatly hugged by Mrs. Ja
mes Dillingham Young, already introduced to you as Della. Which is all very good.\n\nDella finished her cry and attended
to her cheeks with the powder rag. She stood by the window and looked out dully at a grey cat walking a grey fence in a g
rey backyard. To-morrow would be Christmas Day, and she had only $1.87 with which to buy Jim a present. She had been savi
ng every penny she could for months, with this result. Twenty dollars a week doesn\'t go far. Expenses had been greater t
han she had calculated. They always are. Only $1.87 to buy a present for Jim. Her Jim. Many a happy hour she had spent pl
```

In [8]:

```
Tokenizer=nltk.tokenize.TreebankWordTokenizer()
tokens=Tokenizer.tokenize(m)
print(len(tokens))
```

```
2324
```

In [9]:

```
Tokenizer=nltk.tokenize.WhitespaceTokenizer()
tokens=Tokenizer.tokenize(m)
print(len(tokens))
```

```
2074
```

In [10]:

```
Tokenizer=nltk.tokenize.WordPunctTokenizer()
tokens=Tokenizer.tokenize(m)
print(len(tokens))
```

```
2517
```

2.

In [11]:

```
from collections import Counter as c
y=m.split()
count=c(y)
spl=count.most_common(20)
print((spl))
```

```
[('the', 107), ('and', 74), ('a', 64), ('of', 51), ('to', 41), ('was', 26), ('she', 25), ('in', 24), ('had', 21), ('her', 2
1), ('that', 20), ('it', 19), ('at', 19), ('with', 19), ('for', 19), ('his', 17), ('on', 16), ('I', 14), ('Jim', 13), ('wer
e', 11)]
```

3

In [12]:

```
g=set(y)
print(len(g))
```

```
956
```

4

In [13]:

```
import re
w=re.findall(r"\b\w{10,15}\b",m)
g=list(set(w))
print(len(g),g)
```

```
48 ['instigates', 'possessions', 'grandfather', 'comforting', 'Everywhere', 'laboriously', 'hysterical', 'bulldozing', 'int
roduced', 'duplication', 'proclaiming', 'difference', 'mendicancy', 'conception', 'expression', 'wonderfully', 'tremendou
s', 'imputation', 'meretricious', 'mathematician', 'necessitating', 'intoxication', 'appertaining', 'calculated', 'sacrific
ed', 'brilliantly', 'predominating', 'critically', 'sentiments', 'inconsequential', 'reflection', 'adornments', 'depreciat
e', 'contracting', 'ornamentation', 'generosity', 'disapproval', 'possession', 'unassuming', 'worshipped', 'longitudinal',
'uneventful', 'Dillingham', 'description', 'employment', 'prosperity', 'ransacking', 'illuminated']
```

5.

In [14]:

```
from nltk import *
```

In [15]:

```
fw=FreqDist(tokens)
for k,o in fw.items():
    if len(k)>10 and (o)>=2:
        print(k,o)
```

```
description 2
wonderfully 2
```

Exercise-3

In [16]:

```
nltk.download('punkt')
```

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

Out[16]:

```
True
```

In [17]:

```
file2=open("C:\\Users\\user\\Downloads\\austen-emma.txt","r")
f2=file2.read()
f2[-200:]
```

Out[17]:

```
'e deficiencies, the wishes,\nthe hopes, the confidence, the predictions of the small band\nof true friends who witnessed t
he ceremony, were fully answered\nin the perfect happiness of the union.\n\n\nFINIS\n'
```

In [18]:

```
etoks=nltk.word_tokenize(f2.lower())
etoks[-20:]
```

Out[18]:

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

In [19]:

```
len(etoks)
```

Out[19]:

```
191772
```

In [20]:

```
etype=sorted(set(etoks))
```

In [21]:

```
etype[-10:]
```

Out[21]:

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

In [22]:

```
len(etype)
```

Out[22]:

```
7947
```

In [23]:

```
efreq=nltk.FreqDist(etoks)
efreq['beautiful']
```

Out[23]:

```
24
```

step 2

In [24]:

```
#1. Word with prefix and suffix


r=[word for word in etoks if word.startswith("un")& word.endswith("able")]
print(r)
```

```
['unexceptionable', 'unsuitable', 'unreasonable', 'unreasonable', 'uncomfortable', 'unfavourable', 'unexceptionable', 'unex
ceptionable', 'uncomfortable', 'unpersuadable', 'unavoidable', 'unreasonable', 'uncomfortable', 'unsuitable', 'unmanageabl
e', 'unexceptionable', 'unreasonable', 'unobjectionable', 'unpersuadable', 'unsuitable', 'unreasonable', 'uncomfortable',
'unexceptionable', 'unpardonable', 'unmanageable', 'unanswerable', 'unfavourable', 'unpersuadable', 'unaccountable', 'undes
irable', 'unable', 'unable', 'unpardonable', 'unexceptionable', 'unreasonable', 'unreasonable', 'uncomfortable', 'unreasona
ble', 'unpardonable', 'unaccountable', 'unexceptionable', 'unreasonable', 'unaccountable']
```

In [25]:

```
#2. Length

tokenizer=nltk.tokenize.WordPunctTokenizer()
toke=tokenizer.tokenize(f2)
```

In [26]:

```
print(len([word for word in toke if len(word)>15]))
```

```
13
```

In [27]:

```
#Avarage word length

avg=sum(len(word)for word in toke)/len(toke)
avg
```

Out[27]:

```
3.755268231589122
```

In [28]:

```
#4. Word frequency

from nltk import *
fdiemm=FreqDist(toke)
```

In [29]:

```
for i,j in fdiemm.items():
    if j>200:
        print(i,j)
```

```
to 5183
some 248
of 4279
the 4844
; 2199
had 1606
- 574
one 413
in 2118
very 1151
little 354
or 490
her 2381
. 6928
She 562
was 2385
most 243
father 207
' 1007
s 933
```

step3

In [30]:

```
#5. Emma words not in list

e2grm=list(nltk.bigrams(etoks))
e2grm
```

Out[30]:

```
[('[', 'emma'),
 ('emma', 'by'),
 ('by', 'jane'),
 ('jane', 'austen'),
 ('austen', '1816'),
 ('1816', ']'),
 (']', 'volume'),
 ('volume', 'i'),
 ('i', 'chapter'),
 ('chapter', 'i'),
 ('i', 'emma'),
 ('emma', 'woodhouse'),
 ('woodhouse', ','),
 (',', 'handsome'),
 ('handsome', ','),
 (',', 'clever'),
 ('clever', ','),
 ('.', 'and'),
```

In [31]:

```
e2fd=nltk.FreqDist(e2grm)
e2fd
```

Out[31]:

```
FreqDist({(',', 'and'): 1882, ('.', "''"): 1158, ("''", '``'): 958, (';', 'and'): 867, ('to', 'be'): 605, (',', "''"): 584,
('.', 'i'): 569, (',', 'i'): 569, ('of', 'the'): 559, ('in', 'the'): 445, ...})
```

In [32]:

```
#6. last 10 Bigrams

l = FreqDist(dict(e2fd.most_common()[-10:]))
l
```

Out[32]:

```
FreqDist({('who', 'witnessed'): 1, ('witnessed', 'the'): 1, ('the', 'ceremony'): 1, ('were', 'fully'): 1, ('fully', 'answer
ed'): 1, ('answered', 'in'): 1, ('the', 'perfect'): 1, ('the', 'union'): 1, ('union', '.'): 1, ('.', 'finis'): 1})
```

In [33]:

```
#7. Top 20 most frequent bigrams

tokenizer = nltk.tokenize.WhitespaceTokenizer()
tokens = tokenizer.tokenize(f2)
e2grm = list(nltk.bigrams(tokens))
e2fd = nltk.FreqDist(e2grm)
e2fd.most_common(20)
```

Out[33]:

```
[(('to', 'be'), 562),
 (('of', 'the'), 556),
 (('in', 'the'), 431),
 (('I', 'am'), 302),
 (('had', 'been'), 299),
 (('could', 'not'), 270),
 (('it', 'was'), 253),
 (('she', 'had'), 242),
 (('to', 'the'), 236),
 (('have', 'been'), 233),
 (('of', 'her'), 230),
 (('I', 'have'), 214),
 (('and', 'the'), 208),
 (('would', 'be'), 208),
 (('she', 'was'), 206),
 (('do', 'not'), 196),
 (('of', 'his'), 182),
 (('that', 'she'), 178),
 (('to', 'have'), 176),
 (('such', 'a'), 176)]
```

In [34]:

```
#8. Bigram frequency count

for i,j in e2fd.items():
    if i == ('so', 'happy'):
        print(i,j)
```

```
('so', 'happy') 3
```

In [35]:

```
#8. Word Following ( so )

import re
from collections import Counter
words = re.findall(r'so+ \w+',open("C:\\Users\\user\\Downloads\\austen-emma.txt").read())
y = Counter(zip(words))
print(y)
```

```
Counter({('so much',): 95, ('so very',): 76, ('so well',): 30, ('so many',): 27, ('so long',): 27, ('so little',): 20, ('so
far',): 17, ('so I',): 14, ('so kind',): 13, ('so good',): 12, ('so often',): 10, ('so soon',): 9, ('so great',): 8, ('so t
o',): 7, ('so fond',): 7, ('so she',): 7, ('so it',): 6, ('so anxious',): 6, ('so as',): 6, ('so you',): 6, ('so truly',):
6, ('so completely',): 5, ('so obliging',): 5, ('so extremely',): 5, ('so entirely',): 4, ('so happy',): 4, ('so interestin
g',): 4, ('so fast',): 4, ('so near',): 4, ('so pleased',): 4, ('so few',): 4, ('so that',): 4, ('so strong',): 4, ('so lib
eral',): 4, ('so miserable',): 4, ('so happily',): 3, ('so proper',): 3, ('so pleasantly',): 3, ('so superior',): 3, ('so w
armly',): 3, ('so bad',): 3, ('so odd',): 3, ('so ill',): 3, ('so delighted',): 3, ('so particularly',): 3, ('so easily',):
3, ('so on',): 3, ('so attentive',): 3, ('so fortunate',): 3, ('so glad',): 3, ('so shocked',): 3, ('so at',): 3, ('so obli
ged',): 2, ('so perfectly',): 2, ('so dear',): 2, ('so busy',): 2, ('so did',): 2, ('so forth',): 2, ('so totally',): 2,
('so remarkably',): 2, ('so plainly',): 2, ('so charming',): 2, ('so surprized',): 2, ('so early',): 2, ('so too',): 2, ('s
o easy',): 2, ('so decidedly',): 2, ('so absolutely',): 2, ('so particular',): 2, ('so deceived',): 2, ('so palpably',): 2,
('so clever',): 2, ('so short',): 2, ('so cold',): 2, ('so high',): 2, ('so happened',): 2, ('so full',): 2, ('so thoroughl
y',): 2, ('so equal',): 2, ('so off',): 2, ('so naturally',): 2, ('so afraid',): 2, ('so deep',): 2, ('so kindly',): 2, ('s
o pale',): 2, ('so noble',): 2, ('so lovely',): 2, ('so mad',): 2, ('so nearly',): 2, ('so sorry',): 2, ('so cheerful',):
2, ('so unfeeling',): 2, ('so ready',): 2, ('so unperceived',): 1, ('so mild',): 1, ('so constantly',): 1, ('so comfortabl
y',): 1, ('so avowed',): 1, ('so deservedly',): 1, ('so convenient',): 1, ('so just',): 1, ('so apparent',): 1, ('so sorrow
ful',): 1, ('so spent',): 1, ('so artlessly',): 1, ('so plain',): 1, ('so firmly',): 1, ('so genteel',): 1, ('so _then_',):
1, ('so brilliant',): 1, ('so seldom',): 1, ('so nervous',): 1, ('so indeed',): 1, ('so pack',): 1, ('so doubtful',): 1,
('so with',): 1, ('so contemptible',): 1, ('so slightingly',): 1, ('so by',): 1, ('so loudly',): 1, ('so materially',): 1,
('so hard',): 1, ('so delightful',): 1, ('so pointed',): 1, ('so equalled',): 1, ('so evidently',): 1, ('so immediately',):
1, ('so sought',): 1, ('so excellent',): 1, ('so prettily',): 1, ('so extreme',): 1, ('so wonder',): 1, ('so always',): 1,
('so silly',): 1, ('so satisfied',): 1, ('so smiling',): 1, ('so prosing',): 1, ('so undistinguishing',): 1, ('so apt',):
1, ('so dreadful',): 1, ('so respected',): 1, ('so tenderly',): 1, ('so grieved',): 1, ('so shocking',): 1, ('so conceite
d',): 1, ('so before',): 1, ('so prevalent',): 1, ('so heavy',): 1, ('so swiftly',): 1, ('so spoken',): 1, ('so or',): 1,
('so overcharged',): 1, ('so pleasant',): 1, ('so fenced',): 1, ('so hospitable',): 1, ('so interested',): 1, ('so sanguin
e',): 1, ('so sure',): 1, ('so careless',): 1, ('so rapidly',): 1, ('so frequent',): 1, ('so sensible',): 1, ('so misle
d',): 1, ('so blind',): 1, ('so complaisant',): 1, ('so misinterpreted',): 1, ('so active',): 1, ('so pointedly',): 1, ('so
striking',): 1, ('so sudden',): 1, ('so industriously',): 1, ('so partial',): 1, ('so natural',): 1, ('so inevitable',): 1,
('so lately',): 1, ('so beautifully',): 1, ('so distinct',): 1, ('so considerate',): 1, ('so light',): 1, ('so intimate',):
1, ('so magnified',): 1, ('so cautious',): 1, ('so confined',): 1, ('so wish',): 1, ('so he',): 1, ('so glorious',): 1, ('s
o quick',): 1, ('so sweetly',): 1, ('so inseparably',): 1, ('so deserving',): 1, ('so disappointed',): 1, ('so ended',): 1,
('so sluggish',): 1, ('so amiable',): 1, ('so quiet',): 1, ('so idolized',): 1, ('so cried',): 1, ('so acceptable',): 1,
('so properly',): 1, ('so reasonable',): 1, ('so delightfully',): 1, ('so rich',): 1, ('so warm',): 1, ('so large',): 1,
('so handsomely',): 1, ('so abundant',): 1, ('so outree',): 1, ('so thoughtful',): 1, ('so must',): 1, ('so effectually',):
1, ('so beautiful',): 1, ('so Patty',): 1, ('so honoured',): 1, ('so close',): 1, ('so imprudent',): 1, ('so limited',): 1,
('so from',): 1, ('so amusing',): 1, ('so indifferent',): 1, ('so indignant',): 1, ('so said',): 1, ('so right',): 1, ('so
wretched',): 1, ('so now',): 1, ('so occupied',): 1, ('so unhappy',): 1, ('so highly',): 1, ('so generally',): 1, ('so exac
tly',): 1, ('so double',): 1, ('so secluded',): 1, ('so regular',): 1, ('so determined',): 1, ('so motherly',): 1, ('so th
e',): 1, ('so glibly',): 1, ('so calculated',): 1, ('so thrown',): 1, ('so exclusively',): 1, ('so disgustingly',): 1, ('so
needlessly',): 1, ('so does',): 1, ('so resolutely',): 1, ('so would',): 1, ('so infinitely',): 1, ('so fluently',): 1, ('s
o they',): 1, ('so impatient',): 1, ('so briskly',): 1, ('so vigorously',): 1, ('so young',): 1, ('so hardened',): 1, ('so
gratified',): 1, ('so received',): 1, ('so then',): 1, ('so and',): 1, ('so gratefully',): 1, ('so found',): 1, ('so place
d',): 1, ('so lain',): 1, ('so his',): 1, ('so arranged',): 1, ('so moving',): 1, ('so walking',): 1, ('so when',): 1, ('so
favourable',): 1, ('so late',): 1, ('so silent',): 1, ('so dull',): 1, ('so irksome',): 1, ('so agitated',): 1, ('so bruta
l',): 1, ('so cruel',): 1, ('so depressed',): 1, ('so no',): 1, ('so justly',): 1, ('so astonished',): 1, ('so will',): 1,
('so simple',): 1, ('so dignified',): 1, ('so suddenly',): 1, ('so a',): 1, ('so herself',): 1, ('so peremptorily',): 1,
('so uneasy',): 1, ('so wonderful',): 1, ('so _very_',): 1, ('so expressly',): 1, ('so angry',): 1, ('so anxiously',): 1,
('so strange',): 1, ('so stoutly',): 1, ('so mistake',): 1, ('so mistaken',): 1, ('so dreadfully',): 1, ('so voluntaril
y',): 1, ('so satisfactory',): 1, ('so disinterested',): 1, ('so foolishly',): 1, ('so ingeniously',): 1, ('so entreate
d',): 1, ('so like',): 1, ('so cordially',): 1, ('so essential',): 1, ('so designedly',): 1, ('so hasty',): 1, ('so richl
y',): 1, ('so grateful',): 1, ('so tenaciously',): 1, ('so feeling',): 1, ('so engaging',): 1, ('so engaged',): 1, ('so ho
t',): 1, ('so useful',): 1, ('so attached',): 1, ('so peculiarly',): 1, ('so singularly',): 1, ('so taken',): 1, ('so recen
tly',): 1, ('so fresh',): 1, ('so hateful',): 1, ('so heartily',): 1, ('so steady',): 1, ('so complete',): 1, ('so in',):
1, ('so suffered',): 1})
```

In [36]:

```
#10. Trigrams

lto10= FreqDist(dict(e2fd.most_common()[-10:]))
lto10
```

Out[36]:

```
FreqDist({('witnessed', 'the'): 1, ('the', 'ceremony,'): 1, ('ceremony,', 'were'): 1, ('were', 'fully'): 1, ('fully', 'answ
ered'): 1, ('answered', 'in'): 1, ('the', 'perfect'): 1, ('perfect', 'happiness'): 1, ('the', 'union.'): 1, ('union.', 'FIN
IS'): 1})
```

In [37]:

```
#11. Top frequency

e2fd.most_common(10)
```

Out[37]:

```
[(('to', 'be'), 562),
 (('of', 'the'), 556),
 (('in', 'the'), 431),
 (('I', 'am'), 302),
 (('had', 'been'), 299),
 (('could', 'not'), 270),
 (('it', 'was'), 253),
 (('she', 'had'), 242),
 (('to', 'the'), 236),
 (('have', 'been'), 233)]
```

In [38]:

```
#12. Trigram frequency count

e3grm=list(nltk.trigrams(etoks))
e3grm
```

Out[38]:

```
[('[', 'emma', 'by'),
 ('emma', 'by', 'jane'),
 ('by', 'jane', 'austen'),
 ('jane', 'austen', '1816'),
 ('austen', '1816', ']'),
 ('1816', ']', 'volume'),
 (']', 'volume', 'i'),
 ('volume', 'i', 'chapter'),
 ('i', 'chapter', 'i'),
 ('chapter', 'i', 'emma'),
 ('i', 'emma', 'woodhouse'),
 ('emma', 'woodhouse', ','),
 ('woodhouse', ',', 'handsome'),
 (',', 'handsome', ','),
 ('handsome', ',', 'clever'),
 (',', 'clever', ','),
 ('clever', ',', 'and'),
 ('and', 'rich'),
```

In [39]:

```
egramfd = nltk.FreqDist(e3grm)
egramfd
for i,j in egramfd.items():
    if i == ('so', 'happy'):
        print(i,j)
```

In [40]:

```
print(i,j)
```

```
('union', '.', 'finis') 1
```