Taller 9

Métodos Computacionales para Políticas Públicas - URosario

Entrega: viernes 13-nov-2020 11:59 PM

[Ivonne Paola Ubaque Galán]

[ivonne.ubaque@urosario.edu.co (mailto:ivonne.ubaque@urosario.edu.co)]

Instrucciones:

- Guarde una copia de este *Jupyter Notebook* en su computador, idealmente en una carpeta destinada al material del curso.
- Modifique el nombre del archivo del *notebook*, agregando al final un guión inferior y su nombre y apellido, separados estos últimos por otro guión inferior. Por ejemplo, mi *notebook* se llamaría: mcpp_taller9_santiago_matallana
- Marque el *notebook* con su nombre y e-mail en el bloque verde arriba. Reemplace el texto "[Su nombre acá]" con su nombre y apellido. Similar para su e-mail.
- Desarrolle la totalidad del taller sobre este *notebook*, insertando las celdas que sea necesario debajo de cada pregunta. Haga buen uso de las celdas para código y de las celdas tipo *markdown* según el caso.
- Recuerde salvar periódicamente sus avances.
- Cuando termine el taller:
  1. Descárguelo en PDF. Si tiene algún problema con la conversión, descárguelo en HTML.
  2. Suba todos los archivos a su repositorio en GitHub, en una carpeta destinada exclusivamente para este taller, antes de la fecha y hora límites.

NLTK Book (http://www.nltk.org/book/ (http://www.nltk.org/book/)), ejercicios:

- Capítulo 1: 22, 26, 28
- Capítulo 2: 2, 4, 11

CAPITULO UNO

In [10]:
```
#previo instalar: pip install nltk

import nltk
import re
import requests
#nltk.download() #descargar libreria para book
from nltk.book import *
```

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.

In [11]:
```
palabras = [w for w in text5 if len(w) == 4]

#print (sorted(palabras))

print("Lista de palabras mayores a 4 letras")

frecuencia = FreqDist(palabras)
print(frecuencia)
print("")
print("LAS QUE MAS SE REPITEN")

print(frecuencia.most_common(100))
```

```
Lista de palabras mayores a 4 letras
<FreqDist with 1181 samples and 10204 outcomes>

LAS QUE MAS SE REPITEN
[('JOIN', 1021), ('PART', 1016), ('that', 274), ('what', 183), ('here', 181),
('....', 170), ('have', 164), ('like', 156), ('with', 152), ('chat', 142), ('yo
ur', 137), ('good', 130), ('just', 125), ('lmao', 107), ('know', 103), ('room',
98), ('from', 92), ('this', 86), ('well', 81), ('back', 78), ('hiya', 78), ('th
ey', 77), ('dont', 75), ('yeah', 75), ('want', 71), ('love', 60), ('guys', 58),
('some', 58), ('been', 57), ('talk', 56), ('nice', 52), ('time', 50), ('when',
48), ('haha', 44), ('make', 44), ('girl', 43), ('need', 43), ('U122', 42), ('MO
DE', 41), ('will', 40), ('much', 40), ('then', 40), ('over', 39), ('work', 38),
('were', 38), ('take', 37), ('U121', 36), ('U115', 36), ('song', 36), ('even',
35), ('does', 35), ('seen', 35), ('U156', 35), ('U105', 35), ('more', 34), ('da
mn', 34), ('only', 33), ('come', 33), ('hell', 29), ('long', 28), ('them', 28),
('name', 27), ('tell', 27), ('away', 26), ('sure', 26), ('look', 26), ('baby',
26), ('call', 26), ('play', 25), ('U110', 25), ('U114', 25), ('NICK', 24), ('do
wn', 24), ('cool', 24), ('sexy', 23), ('many', 23), ('hate', 23), ('said', 23),
('last', 22), ('ever', 22), ('hear', 21), ('life', 21), ('live', 20), ('feel',
19), ('very', 19), ('mean', 19), ('give', 19), ('same', 19), ('must', 19), ('st
op', 19), ('LMAO', 19), ('!!!!', 18), ('hugs', 18), ('What', 18), ('find', 18),
('cant', 18), ('left', 17), ('????', 17), ('shit', 17), ('nite', 17)]
```

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?

In [3]:
```
print(sum(len(w) for w in text5))

print("R/ No, este codigo sirve para mumar la cantidad total de caracteres con la
```

```
158114
R/ No, este codigo sirve para mumar la cantidad total de caracteres con la cual
es esta conformado el texto.
```

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.

In [4]:
```
def porcent(palabra,text):
    mitexto = [w for w in text if len(w) > 2] # las palabras se forman a partir d
    total= len(mitexto) #tamaño al estar separado por palabras contamos la cantid
    numero = mitexto.count(palabra) #busca la cantidad de la palabra que estamos
    res = (numero/total)*100 # operacion para hallar porcentaje
    print ("La palabra ",palabra,"representa el ",round(res,3),"%")
```

In [5]:
```
palabra = input("Ingrese la palabra que desa buscar en Text5:")
porcent(palabra,text5)

#Buscar la palabra, para este caso JOIN
```

```
Ingrese la palabra que desa buscar en Text5:JOIN
La palabra  JOIN representa el  3.332 %
```

CAPITULO 2

In [6]:
```
nltk.corpus.gutenberg.fileids()
```

Out[6]:
```
['austen-emma.txt',
 'austen-persuasion.txt',
 'austen-sense.txt',
 'bible-kjv.txt',
 'blake-poems.txt',
 'bryant-stories.txt',
 'burgess-busterbrown.txt',
 'carroll-alice.txt',
 'chesterton-ball.txt',
 'chesterton-brown.txt',
 'chesterton-thursday.txt',
 'edgeworth-parents.txt',
 'melville-moby_dick.txt',
 'milton-paradise.txt',
 'shakespeare-caesar.txt',
 'shakespeare-hamlet.txt',
 'shakespeare-macbeth.txt',
 'whitman-leaves.txt']
```

2 Use the corpus module to explore austen-persuasion.txt. How many word tokens does this book have? How many word types?

In [7]:
```
palabras = nltk.corpus.gutenberg.words('austen-persuasion.txt')

print("Contiene ",len(palabras)," palabras y",len(set(palabras)),"tipos de palabr
```

```
Contiene  98171  palabras y 6132 tipos de palabras.
```

4 Read in the texts of the State of the Union addresses, using the state_union corpus reader. Count occurrences of men, women, and people in each document. What has happened to the usage of these words over time?

In [8]:
```
textos = nltk.corpus.state_union.fileids() # asignamos los textos a la variable
#print(textos) #imprimimos para saber que contiene

resultado = [None]*3
buscar=['men','women','people']
contenido_p = []
for texto in nltk.corpus.state_union.fileids(): # usamos el for para separar los

    for palabra in nltk.corpus.state_union.words(texto):
         contenido_p.append(palabra)

    for i in range(3):
        numero = contenido_p.count(buscar[i])
        print(texto, buscar[i],": ",numero)
```

```
1955-Eisenhower.txt men :  50
1955-Eisenhower.txt women :  17
1955-Eisenhower.txt people :  190
1956-Eisenhower.txt men :  52
1956-Eisenhower.txt women :  19
1956-Eisenhower.txt people :  220
1957-Eisenhower.txt men :  57
1957-Eisenhower.txt women :  21
1957-Eisenhower.txt people :  231
1958-Eisenhower.txt men :  59
1958-Eisenhower.txt women :  22
1958-Eisenhower.txt people :  250
1959-Eisenhower.txt men :  63

1959-Eisenhower.txt women :  23
1959-Eisenhower.txt people :  261
1960-Eisenhower.txt men :  65
1960-Eisenhower.txt women :  23
1960-Eisenhower.txt people :  271
```

El texto sugiere que antes de 1970 el término mujer era menos usada, pero desde entonces se ha usado más o menos como el término hombres . Personas es el término mas usado, especialmente en los últimos años.

11 Investigate the table of modal distributions and look for other patterns. Try to explain them in terms of your own impressionistic understanding of the different genres. Can you find other closed classes of words that exhibit significant differences across different genres?

In [9]:
```
from nltk.corpus import brown
texto = nltk.ConditionalFreqDist(
    (genero,palabra)
    for genero in brown.categories()
    for palabra in brown.words(categories=genero))

generos = ['adventure', 'government', 'hobbies', 'mystery', 'belles_lettres', 're

palabras = ["who", "might", "when", "where", "will", "how"]

texto.tabulate(conditions=generos, samples=palabras)
# most frequent in new is who, when;religion is who, what;hobbies is who, when,et


rta2 = "Rta - Si, se puede encontrar clases cerradas de palabras"

print(rta2)
```

```
                who might  when where  will   how
     adventure   91    58   126    53    50    35
    government   74    13    56    46   244    16
       hobbies  103    22   119    72   264    40
       mystery   80    57   114    59    20    37
belles_lettres  452   113   252   107   236    96
       reviews  128    26    54    25    58    26
Rta - Si, se puede encontrar clases cerradas de palabras
```