Taller 9

Métodos Computacionales para Políticas Públicas - URosario

Entrega: viernes 1-nov-2019 11:59 PM

Juan Sebastián Muñoz

jsebastianmvargas@gmail.com (mailto:jsebastianmvargas@gmail.com)

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/)), Exercises:

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

Chapter 1

In [13]:
```
## Importar las librerias a usar
import nltk
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"] = [18.0, 8.0]
```

In [2]:
```
nltk.download("book")
```

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

[nltk_data]    |     C:\Users\User\AppData\Roaming\nltk_data...
[nltk_data]    |   Package book_grammars is already up-to-date!
[nltk_data]    | Downloading package city_database to
[nltk_data]    |     C:\Users\User\AppData\Roaming\nltk_data...
[nltk_data]    |   Package city_database is already up-to-date!
[nltk_data]    | Downloading package tagsets to
[nltk_data]    |     C:\Users\User\AppData\Roaming\nltk_data...
[nltk_data]    |   Package tagsets is already up-to-date!
[nltk_data]    | Downloading package panlex_swadesh to
[nltk_data]    |     C:\Users\User\AppData\Roaming\nltk_data...
[nltk_data]    |   Package panlex_swadesh is already up-to-date!
[nltk_data]    | Downloading package averaged_perceptron_tagger to
[nltk_data]    |     C:\Users\User\AppData\Roaming\nltk_data...
[nltk_data]    |   Package averaged_perceptron_tagger is already up-
```

In [3]:
```
from nltk.book import *
```

```
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
```

Ejercicios 22, 26, 28

In [4]:
```
#Ejercicio 22

#Find all the four-letter words in the Chat Corpus (text5). With the help of a fr
#show these words in decreasing order of frequency

Four_letter_words = [w for w in text5 if len(w) == 4]
Four_letter_words
Flw = FreqDist(Four_letter_words)
Flw
```

Out[4]:
```
FreqDist({'JOIN': 1021, 'PART': 1016, 'that': 274, 'what': 183, 'here': 181,
'....': 170, 'have': 164, 'like': 156, 'with': 152, 'chat': 142, ...})
```

In [5]:
```
#Ejercicio 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?
sum(len(w) for w in text1)
# El código anterior suma la longitud de todas las palabras del texto 1. Por lo c
#de palabras, al dividir por la cantidad de palabras
print("Este es el promedio de longitud de las palabras en el texto 1: " , sum(le
```

```
Este es el promedio de longitud de las palabras en el texto 1:  3.8304111280236
49
```

In [7]:
```
#Ejercicio 28
#Define a function percent(word, text) that calculates how often a given word oc
#and expresses the result as a percentage
def percent(word, text):
    return str(100 * text.count(word) / len(text))+"%"
percent("Moby", text1)
```

Out[7]:
```
'0.032206242643365704%'
```

Chapter 2

Ejercicios 2, 4, 11

In [8]:
```
nltk.corpus.gutenberg.fileids()
```

Out[8]:
```
['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']
```

In [9]:
```
#Ejercicio 2
#Use the corpus module to explore austen-persuasion.txt.
#How many word tokens does this book have? How many word types?
pers = nltk.corpus.gutenberg.words('austen-persuasion.txt')
pers_alpha = [w for w in pers if w.isalpha()]
print ("Cantidad de tokens de palabras:" , len(pers_alpha))
print ("Cantidad de tokens de palabras únicas:" , len(set(pers_alpha)))
```

```
Cantidad de tokens de palabras: 84121
Cantidad de tokens de palabras únicas: 6036
```

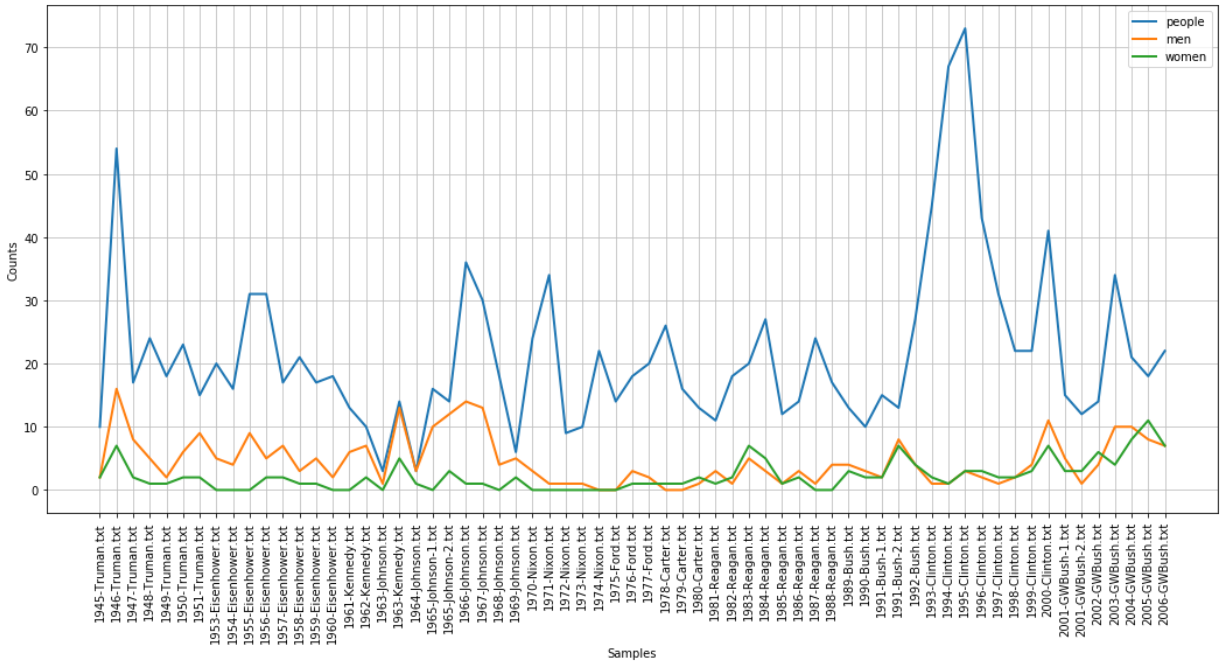
In [10]:
```
#Ejercicio 4
#Read in the texts of the State of the Union addresses, using the state_union co
#Count occurrences of men, women, and people in each document.
#What has happened to the usage of these words over time?
```

In [11]:
```
nltk.corpus.state_union.fileids()
```

Out[11]:
```
['1945-Truman.txt',
 '1946-Truman.txt',
 '1947-Truman.txt',
 '1948-Truman.txt',
 '1949-Truman.txt',
 '1950-Truman.txt',
 '1951-Truman.txt',
 '1953-Eisenhower.txt',
 '1954-Eisenhower.txt',
 '1955-Eisenhower.txt',
 '1956-Eisenhower.txt',
 '1957-Eisenhower.txt',
 '1958-Eisenhower.txt',
 '1959-Eisenhower.txt',
 '1960-Eisenhower.txt',
 '1961-Kennedy.txt',
 '1962-Kennedy.txt',
 '1963-Johnson.txt',
 '1963-Kennedy.txt',
```

In [14]:
```
cfd = nltk.ConditionalFreqDist(
    (target, fileid[:])
    for fileid in nltk.corpus.state_union.fileids()
    for w in nltk.corpus.state_union.words(fileid)
    for target in ['men', 'women', 'people']
    if w.lower().startswith(target))

cfd.plot();
```



In [15]:
```
#Ejercicio 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 d
#Can you find other closed classes of words that exhibit significant differences
from nltk.corpus import brown
cfd = nltk.ConditionalFreqDist(
    (genre, word)
    for genre in brown.categories()
    for word in brown.words(categories=genre))
genres = ['news', 'religion', 'hobbies', 'science_fiction', 'romance', 'humor']
modals = ['can', 'could', 'may', 'might', 'must', 'will']
cfd.tabulate(conditions=genres, samples=modals)
```

```
                  can could   may might  must  will
           news    93    86    66    38    50   389
       religion    82    59    78    12    54    71
        hobbies   268    58   131    22    83   264
science_fiction    16    49     4    12     8    16
        romance    74   193    11    51    45    43
          humor    16    30     8     8     9    13
```

Las noticias se destacan por su uso de 'will', destacando aquel carácter de analizar la realidad para poder definir lo que sucederá, dadas las situaciones que reportan. En similitud con lo que sucede con los Hobbies, donde las personas al soñar buscan relatar sus futuras acciones (will) y lo que podrán hacer (can). Asimismo, en el caso del romance, se hace evidente lo importante del verbo 'could' como símbolo de cortesía. En el caso del humor, más allá de la frecuencia particular de cada uno de estos verbos, la frecuencia general del uso de estos verbos modales es tan baj, que refleja la poca complejidad y elaboración del vocabulario en este género.

In [16]:
```
cfd = nltk.ConditionalFreqDist(
    (genre, word)
    for genre in brown.categories()
    for word in brown.words(categories=genre))
genres = ['news', 'religion', 'hobbies', 'science_fiction', 'romance', 'humor']
pronouns = ['I', 'you', 'he', 'she', 'we', 'they']
cfd.tabulate(conditions=genres, samples=pronouns)
```

```
                   I  you   he  she   we they
           news  179   55  451   42   77  205
       religion  155  100  137   10  176  115
        hobbies  154  383  155   21  100  177
science_fiction   98   81  139   36   30   53
        romance  951  456  702  496   78  168
          humor  239  131  146   58   32   70
```

Al revisar sobre que tipo de personas son los principales actores, a través de los distintos géneros, se destacan los cambios de pronombres. En todos los géneros se hace más referencia a un hombre, que a una mujer, resaltando su protagonismo. Algo que es menor en el género de romance, donde la presencia femenina es más latente y, en consecuencia, su participación. Para el caso de las noticias, se destaca que los principales actores son hombres y grupos, seguidos por el pronombre 'I', lo que puede ser dado al uso frecuente de entrevistas. Por otro lado, en cuanto al uso de 'I', se destaca su presencia en los géneros de romance y humor, al ser los principales donde su rol es narrativo y la primera persona adquiere una presencia clave, ya sea porque los

mismos personajes son quienes hablan, o porque es el pronombre usado para contar anéctodas que generarán alguna gracia. En el género de ciencia ficción y en la religion, al igual que en los demás, los principales actores continúan siendo hombres. La carencia de participación femenina destaca principalmente en la religión, donde tiene menos presencia.