Pontificia Universidad Católica de Chile

Facultad de Letras

Introducción a la lingüística computacional

César Antonio Aguilar
Facultad de Lenguas y Letras
14/09/2017

Cesar.Aguilar72@gmail.com

Síntesis de la clase anterior

En la clase anterior, hicimos una exploración dentro del **Corpus Brown**, usando algunas de las funciones que nos ofrecen los códigos de NLTK.

BROWN

Brown Corpus

- 1 million words
- 500 texts
- Varied genres (newspaper, novels, non-fiction, academic, etc.)
- Assembled at Brown University in 1963-64
- The first large on-line text collection used in corpus-based NLP research

Etiquetando documentos

¿Para que nos sirven estos corpus? Precisamente para insertar etiquetas en documentos que no las tienen, p. e.:

```
>>> text = nltk.word_tokenize("They refuse to permit us to obtain the refuse permit")
>>> nltk.pos_tag(text)
[('They', 'PRP'), ('refuse', 'VBP'), ('to', 'TO'), ('permit', 'VB'), ('us', 'PRP'),
('to', 'TO'), ('obtain', 'VB'), ('the', 'DT'), ('refuse', 'NN'), ('permit', 'NN')]
```

Ejercicio (1)

Vamos a etiquetar un documento usando las etiquetas que manera el módulo POS de NLTK. Primero, importemos los siguientes módulos:

```
import nltk, re, urllib
```

Hecho esto, de la librería **urllib** importemos la función **request**:

```
from urllib import request
```

Ejercicio (2)

Como hemos importado la librería **urllib**, podemos procesar un documento obtenido de Internet. Aprovechemos esto para darle un vistazo al **Proyecto Gutenberg**:

www.gutenberg.org

El Proyecto Gutenberg consiste en una biblioteca digital que ofrece acceso gratuito a una colección de aproximadamente 54.000 volúmenes en distintas lenguas.

Ejercicio (3)

Vamos a descargar un libro del Proyecto Gutenberg, en este caso, una novela de John William Polidori, *A Vampire*. Usemos las siguientes líneas:

```
url01 = ‘http://www.gutenberg.org/cache/epub/6087/pg6087.txt’

html01 = request.urlopen(url01).read().decode('utf8')
```

Ocupemos de nuevo Beautiful Soup para trasnformar nuestra secuencia de bytes en una cadena de caracteres:

```
from bs4 import BeautifulSoup
```

Ejercicio (4)

Hagamos nuestra transformación:

```
Cadena_Polidori01 = BeautifulSoup(html02).get_text()
```

Una vez que hemos hecho esto, la historia de Polidori se ha convertido en una cadena de caracteres. Para transformarla en una lista de palabras, requerimos las siguientes instrucciones:

```
from nltk import word_tokenize

Tokens_Polidori01 = word_tokenize(Cadena_Polidori01)
```

Corroboramos:

```
type(Tokens_Polidori01)

len(Tokens_Polidori01)
```

Ejercicio (5)

Ahora, vamos a etiquetar nuestro texto, para lo cual podemos usar la siguiente función:

```
Tagged_Polidori01 = nltk.pos_tag(Tokens_Polidori01)
```

Veamos nuestros resultados:

```
Rango_Tagged_Polidori01 =
Tagged_Polidori01[0:100]

print(Rango_Tagged_Polidori01)
```

Ejercicio (6)

Y obtenemos:

```
In [13]: print(Rango_Tagged_Polidori01)
[('\ufeffThe', 'DT'), ('Project', 'NNP'), ('Gutenberg', 'NNP'), ('EBook', 'NNP'),
('of', 'IN'), ('The', 'DT'), ('Vampyre', 'NNP'), (';', ':'), ('A', 'DT'),
('Tale', 'NNP'), (',', ','), ('by', 'IN'), ('John', 'NNP'), ('William', 'NNP'),
('Polidori', 'NNP'), ('This', 'NNP'), ('eBook', 'NNP'), ('is', 'VBZ'), ('for',
'IN'), ('the', 'DT'), ('use', 'NN'), ('of', 'IN'), ('anyone', 'NN'), ('anywhere',
'RB'), ('at', 'IN'), ('no', 'DT'), ('cost', 'NN'), ('and', 'CC'), ('with', 'IN'),
('almost', 'RB'), ('no', 'DT'), ('restrictions', 'NNS'), ('whatsoever', 'VBP'),
('.', '.'), ('You', 'PRP'), ('may', 'MD'), ('copy', 'VB'), ('it', 'PRP'), (',',
','), ('give', 'VBP'), ('it', 'PRP'), ('away', 'RB'), ('or', 'CC'), ('re-use',
'JJ'), ('it', 'PRP'), ('under', 'IN'), ('the', 'DT'), ('terms', 'NNS'), ('of',
'IN'), ('the', 'DT'), ('Project', 'NNP'), ('Gutenberg', 'NNP'), ('License',
'NNP'), ('included', 'VBD'), ('with', 'IN'), ('this', 'DT'), ('eBook', 'NNP'),
('or', 'CC'), ('online', 'NN'), ('at', 'IN'), ('www.gutenberg.net', 'JJ'),
('Title', 'NN'), (':', ':'), ('The', 'DT'), ('Vampyre', 'NNP'), (';', ':'), ('A',
'DT'), ('Tale', 'NNP'), ('Author', 'NNP'), (':', ':'), ('John', 'NNP'),
('William', 'NNP'), ('Polidori', 'NNP'), ('Posting', 'NNP'), ('Date', 'NNP'),
(':', ':'), ('October', 'NNP'), ('21', 'CD'), (',', ','), ('2009', 'CD'), ('[',
':'), ('EBook', 'NNP'), ('#', '#'), ('6087', 'CD'), (']', 'CD'), ('Release',
'NNP'), ('Date', 'NNP'), (':', ':'), ('July', 'NNP'), (',', ','), ('2004', 'CD'),
('First', 'NNP'), ('Posted', 'NNP'), (':', ':'), ('November', 'NNP'), ('3',
'CD'), (',', ','), ('2002', 'CD'), ('[', ':'), ('Last', 'JJ')]
```

Ejercicio (7)

Ahora, generemos un nuevo archivo con extensión TXT que contenga nuestro texto etiquetado:

```
Final_Polidori01 =
open('c://Anaconda3/Corpus_Pruebas/Results_Polidori01.txt', 'w')
for elem in Tagged_Polidori01:
    Final_Polidori01.write(str(elem)+" ")

Final_Polidori01.close()
```

Veamos nuestro resultado:

Ejercicio (8)

Results_Polidori01: Bloc de notas

Archivo Edición Formato Ver Ayuda


```
('\ufeffThe', 'DT') ('Project', 'NNP') ('Gutenberg', 'NNP') ('EBook', 'NNP') ('of', 'IN')
('The', 'DT') ('Vampyre', 'NNP') (';', ':') ('A', 'DT') ('Tale', 'NNP') (',', ',') ('by',
'IN') ('John', 'NNP') ('William', 'NNP') ('Polidori', 'NNP') ('This', 'NNP') ('eBook',
'NNP') ('is', 'VBZ') ('for', 'IN') ('the', 'DT') ('use', 'NN') ('of', 'IN') ('anyone',
'NN') ('anywhere', 'RB') ('at', 'IN') ('no', 'DT') ('cost', 'NN') ('and', 'CC') ('with',
'IN') ('almost', 'RB') ('no', 'DT') ('restrictions', 'NNS') ('whatsoever', 'VBP') ('.',
'.') ('You', 'PRP') ('may', 'MD') ('copy', 'VB') ('it', 'PRP') (',', ',') ('give', 'VBP')
('it', 'PRP') ('away', 'RB') ('or', 'CC') ('re-use', 'JJ') ('it', 'PRP') ('under', 'IN')
('the', 'DT') ('terms', 'NNS') ('of', 'IN') ('the', 'DT') ('Project', 'NNP')
('Gutenberg', 'NNP') ('License', 'NNP') ('included', 'VBD') ('with', 'IN') ('this', 'DT')
('eBook', 'NNP') ('or', 'CC') ('online', 'NN') ('at', 'IN') ('www.gutenberg.net', 'JJ')
('Title', 'NN') (':', ':') ('The', 'DT') ('Vampyre', 'NNP') (';', ':') ('A', 'DT')
('Tale', 'NNP') ('Author', 'NNP') (':', ':') ('John', 'NNP') ('William', 'NNP')
('Polidori', 'NNP') ('Posting', 'NNP') ('Date', 'NNP') (':', ':') ('October', 'NNP')
('21', 'CD') (',', ',') ('2009', 'CD') ('[', ':') ('EBook', 'NNP') ('#', '#') ('6087',
'CD') (']', 'CD') ('Release', 'NNP') ('Date', 'NNP') (':', ':') ('July', 'NNP') (',',
',') ('2004', 'CD') ('First', 'NNP') ('Posted', 'NNP') (':', ':') ('November', 'NNP')
('3', 'CD') (',', ',') ('2002', 'CD') ('[', ':') ('Last', 'JJ') ('updated', 'VBN') (':',
':') ('May', 'NNP') ('26', 'CD') (',', ',') ('2012', 'CD') (']', ':') ('Language', 'NNP')
(':', ':') ('English', 'JJ') ('***', '-NONE-') ('START', 'NNP') ('OF', 'NNP') ('THIS',
'NNP') ('PROJECT', 'NNP') ('GUTENBERG', 'NNP') ('EBOOK', 'NNP') ('THE', 'NNP')
```

Frecuencia de palabras en corpus (1)

Volviendo con las opciones de exploración que nos ofrecía el Corpus Brown, revisemos otras funciones que nos ayudan a analizar el comportamiento numérico de una colección de textos. Para esto, vamos a usar el **Inaugural Address Corpus**, que es una recopilación de los discursos que han dado los presidentes de Estados Unidos al asumir su cargo. Para esto, necesitamos de la instrucción:

```
from  nltk.corpus import inaugural
```

Veamos el contenido de este corpus con los siguientes comandos:

```
1. inaugural.fileids()

2. [fileid[:4] for fileid in inaugural.fileids()]
```

Frecuencia de palabras en corpus (2)

Primer resultado:

```
>>> from nltk.corpus import inaugural
>>> inaugural.fileids()
['1789-Washington.txt', '1793-Washington.txt', '1797-Adams.txt', '1801-Jefferson
.txt', '1805-Jefferson.txt', '1809-Madison.txt', '1813-Madison.txt', '1817-Monro
e.txt', '1821-Monroe.txt', '1825-Adams.txt', '1829-Jackson.txt', '1833-Jackson.t
xt', '1837-VanBuren.txt', '1841-Harrison.txt', '1845-Polk.txt', '1849-Taylor.txt
', '1853-Pierce.txt', '1857-Buchanan.txt', '1861-Lincoln.txt', '1865-Lincoln.txt
', '1869-Grant.txt', '1873-Grant.txt', '1877-Hayes.txt', '1881-Garfield.txt', '1
885-Cleveland.txt', '1889-Harrison.txt', '1893-Cleveland.txt', '1897-McKinley.tx
t', '1901-McKinley.txt', '1905-Roosevelt.txt', '1909-Taft.txt', '1913-Wilson.txt
', '1917-Wilson.txt', '1921-Harding.txt', '1925-Coolidge.txt', '1929-Hoover.txt'
, '1933-Roosevelt.txt', '1937-Roosevelt.txt', '1941-Roosevelt.txt', '1945-Roosev
elt.txt', '1949-Truman.txt', '1953-Eisenhower.txt', '1957-Eisenhower.txt', '1961
-Kennedy.txt', '1965-Johnson.txt', '1969-Nixon.txt', '1973-Nixon.txt', '1977-Car
ter.txt', '1981-Reagan.txt', '1985-Reagan.txt', '1989-Bush.txt', '1993-Clinton.t
xt', '1997-Clinton.txt', '2001-Bush.txt', '2005-Bush.txt', '2009-Obama.txt']
>>>
```

Segundo resultado:

```
>>> [fileid[:4] for fileid in inaugural.fileids()]
['1789', '1793', '1797', '1801', '1805', '1809', '1813', '1817', '1821', '1825',
 '1829', '1833', '1837', '1841', '1845', '1849', '1853', '1857', '1861', '1865',
 '1869', '1873', '1877', '1881', '1885', '1889', '1893', '1897', '1901', '1905',
 '1909', '1913', '1917', '1921', '1925', '1929', '1933', '1937', '1941', '1945',
 '1949', '1953', '1957', '1961', '1965', '1969', '1973', '1977', '1981', '1985',
 '1989', '1993', '1997', '2001', '2005', '2009']
>>>
```

Frecuencia de palabras en corpus (3)

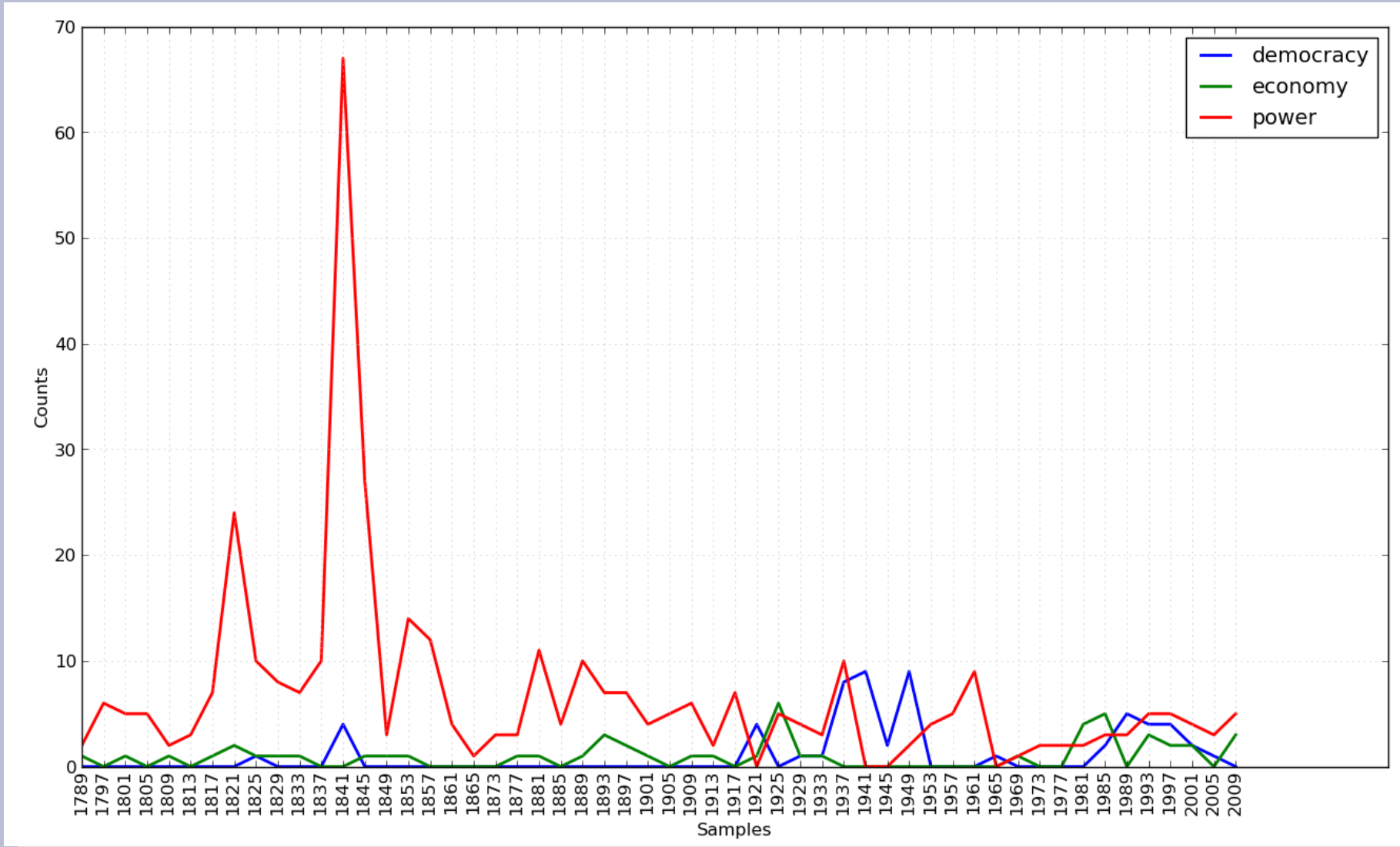
Pasemos entonces a hacer nuestro gráfico, para lo que necesitamos de nuevo la función **Conditional FreqDist**. Del mismo modo, vamos a visualizar la frecuencia del uso de 3 palabras: *democracy*, *economy* y *power*.

```
>>> cfd_inaugural01 = nltk.ConditionalFreqDist(
                        (target, fileid[:4])
                        for fileid in inaugural.fileids()
                        for w in inaugural.words(fileid)
                        for target in ['democracy', 'economy', 'power']
                        if w.lower().startswith(target))

>>> cfd_inaugural01.plot()
```

Veamos el resultado en la siguiente lámina:

Frecuencia de palabras en corpus (4)



Frecuencia de palabras en corpus (5)

¿Qué podemos hacer con nuestro documento convertido en una lista de tokens? De entrada, una primera tarea sería hacer algunos cálculos que nos permitan reconocer algunas propiedades respecto al vocabulario que contienen. Para esto, podemos ocupar el módulo FreqDist, el cual incorpora diferentes funciones que nos permiten analizar el vocabulario de cada Texto, así como la frecuencia de aparición de algún elemento (en orden decreciente), en concreto, cualquier palabra posible.

Vamos a probar este módulo usando la siguiente instrucción:

```
Fdist_Polidori01 = nltk.FreqDist(Tokens_Polidori01)
```

Frecuencia de palabras en corpus (6)

Algunas funciones que podemos ocupar para identificar frecuencias de distribución en nuestros textos son las siguientes:

Ejemplo	Descripción
fdist['Palabra']	count of the number of times a given sample occurred
fdist.freq('Palabra')	frequency of a given sample
fdist.N()	total number of samples
fdist.most_common(Número)	the n most common samples and their frequencies
fdist.max()	sample with the greatest count
fdist.plot(Número)	graphical plot of the frequency distribution
fdist.plot(Número, cumulative=True)	cumulative plot of the frequency distribution

Frecuencia de palabras en corpus (7)

Tratemos de aplicar algunas de las funciones que hemos visto, p. e.: ¿cuáles son las primeras 50 palabras más recurrentes en nuestro texto?:

```
Fdist_Polidori01 = nltk.FreqDist(Tokens_Polidori01)
```

IPython

File Edit View Kernel Magic Window Help

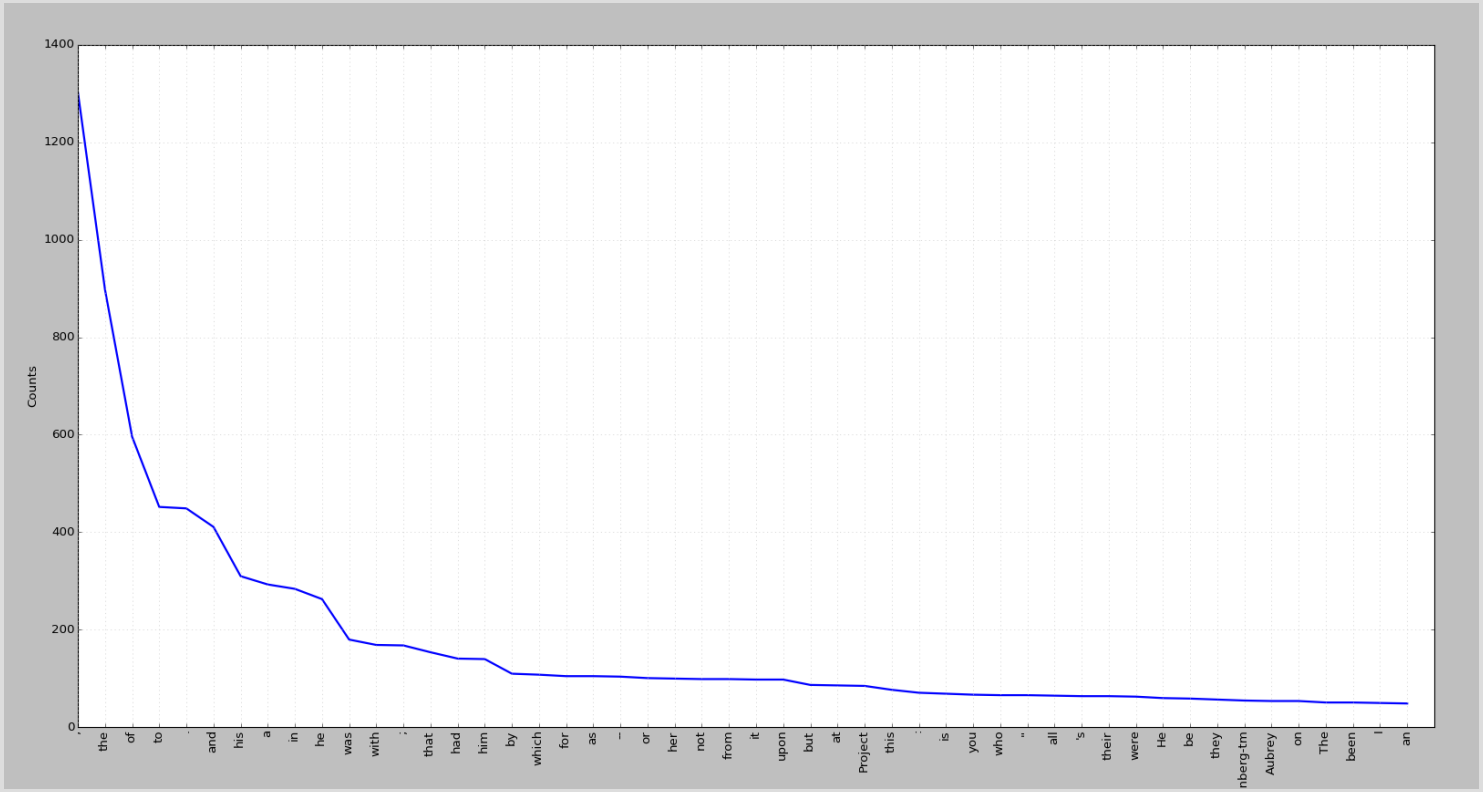
```
In [23]: Fdist_Polidori01.most_common(50)
Out[23]:
[(',', 1303),
 ('the', 898),
 ('of', 596),
 ('to', 452),
 ('.', 449),
 ('and', 411),
 ('his', 310),
 ('a', 293),
 ('in', 284),
 ('he', 263),
 ('was', 180),
 ('with', 169),
 (';', 168),
 ('that', 154),
 ('had', 141),
```

Frecuencia de palabras en corpus (8)

También podemos generar un gráfico con estos mismos datos. Veamos:

Fdist_Polidori01.plot(50)

Obtenemos:



Regreso al módulo *nltk.book* (1)

Finalmente, para recordar que también podemos usar funciones ya vistas del módulo **nltk.book**, transformemos nuestra lista de tokens en un objeto con el atributo **.Text**:

```
Text_Polidori01 = nltk.Text(Tokens_Polidori01)
```

Reconozcamos algunas colocaciones:

```
Text_Polidori01.collocations()
```

Igualmente, podemos generar algunas concordancias:

```
Text_Polidori01.concordance('vampyre')
```

Gracias por su atención

Blog del curso:
http://cesaraguilar.weebly.com/curso-de-procesamiento-del-lenguaje-natural.html