Today's Topic

• May I assume everyone has completed Homework 3?
• Then Homework 4!
import nltk
from nltk.corpus import ptb
t = ptb.words(categories=['news'])
len(t)
1253013
text = nltk.Text(t)
len(text)
1253013
type(text)
<class 'nltk.text.Text'>
words = set(text)
len(words)
49817

len(words)/len(text)
0.03975776787631094 (lexical diversity 4%)
text.count('the')
50975
text.count('The')
8442

text[1000]
'be'
text.index('be')
480

Acknowledgment: adapted from nltk book chapter 1
roughly speaking, the most common bigrams without stopwords
stopwords.words('english')
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'you're', 'you've', 'you'll', 'you'd', 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', 'she's', 'her', 'hers', 'herself', 'it', 'it's', 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', 'that'll', 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'don't', 'should', 'should've', 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', 'aren't', 'couldn', 'couldn't', 'didn', 'didn't', 'doesn', 'doesn't', 'hadn', 'hadn't', 'hasn', 'hasn't', 'haven', 'haven't', 'isn', 'isn't', 'ma', 'mightn', 'mightn't', 'mustn', 'mustn't', 'needn', 'needn't', 'shan', 'shan't', 'shouldn', 'shouldn't', 'wasn', 'wasn't', 'weren', 'weren't', 'won', 'won't', 'wouldn', 'wouldn't']
50 most common words: cumulative freq dist

d = nltk.FreqDist(text)
print(d)
<FreqDist with 49817 samples and 1253013 outcomes>
d.plot(50,cumulative=True)
**Aside:** `plot()` requires `matplotlib` to be installed

```
$ pip3 install matplotlib
Collecting matplotlib
  Downloading matplotlib-2.1.2-cp35-cp35m-macosx_10_6_intel.macosx_10_9_intel.macosx_10_9_x86_64.macosx_10_10_intel.macosx_10_10_x86_64.whl (13.2MB)
    100% |████████████████████████████████| 13.2MB 95kB/s
Collecting pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 (from matplotlib)
  Downloading pyparsing-2.2.0-py2.py3-none-any.whl (56kB)
    100% |████████████████████████████████| 61kB 1.3MB/s
Collecting cycler>=0.10 (from matplotlib)
  Downloading cycler-0.10.0-py2.py3-none-any.whl
Collecting pytz (from matplotlib)
  Downloading pytz-2017.3-py2.py3-none-any.whl (511kB)
    100% |████████████████████████████████| 512kB 1.1MB/s
Requirement already satisfied: six>=1.10 in /Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site-packages (from matplotlib)
Collecting python-dateutil>=2.1 (from matplotlib)
  Downloading python_dateutil-2.6.1-py2.py3-none-any.whl (194kB)
    100% |████████████████████████████████| 194kB 1.3MB/s
Requirement already satisfied: numpy>=1.7.1 in /Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site-packages (from matplotlib)
Installing collected packages: pyparsing, cycler, pytz, python-dateutil, matplotlib
Successfully installed cycler-0.10.0 matplotlib-2.1.2 pyparsing-2.2.0 python-dateutil-2.6.1 pytz-2017.3
```
"U"'s POS tag is -NONE-

text.index('U')
1108
text[1097:1120]
['Assets', 'of', 'the', '400', 'taxable', 'funds', 'grew', 'by', '$', '1.5', 'billion', 'U', 'during', 'the', 'latest', 'week', ',', 'to', '$', '352.7', 'billion', 'U', '].

Delete words tagged as -NONE-
Text corpus (revisited): cumulative freq dist

t2 = [x[0] for x in ptb.tagged_words(categories=['news']) if x[1] != '-NONE-']
len(t2)
1173766
text = nltk.Text(t2)
d2 = nltk.FreqDist(text)
d2.plot(50,cumulative=True)
Text corpus (revisited): cumulative freq dist

# words: 1253013 - 1173766 = 79247 words tagged as -NONE-

set(x[8] for x in pbtag.tagged_words(categories=['\news']) if x[1]==-\NONE-)
Explaining *T*

Who bought a book?
Text corpus punctuation

Punctuation tags (textbook is missing these):
1. –LRB– used for –LCB– (left curly bracket) and –LRB– (left round bracket)
2. –RRB– used for –RCB– (right curly bracket) and –RRB– (right round bracket)
3. SYM used for \{'r', 'g', 'e', 'z', 'd', 'b', 'FFr', '&', '=', 'f', 'c', 'x', 'a', '@', '*\}\
4. : used for {';', '–', ':', '...', '--'}
5. . used for {'.?', '!', '.}
6. , used for ,
7. ` used for “ and`
8. ’ used for ” and’
nltk.FreqDist

- [http://www.nltk.org/_modules/nltk/probability.html#FreqDist](http://www.nltk.org/_modules/nltk/probability.html#FreqDist)
- `dist = nltk.FreqDist(text)`
- `dist.max() => 'the'`
- `dist.hapaxes()`
- `dist.freq('the')`
- `dist.tabulate(10)`
- ', the . of to a and in 's for
- 60484 50975 48144 28338 27249 23673 19762 18857 11539 9890

```
class nltk.probability.FreqDist(samples=None) [source]
    Bases: collections.Counter

    A frequency distribution for the outcomes of an experiment. A frequency distribution records
    the number of times each outcome of an experiment has occurred. For example, a
    frequency distribution could be used to record the frequency of each word type in a
    document. Formally, a frequency distribution can be defined as a function mapping from
    each sample to the number of times that sample occurred as an outcome.
```
Class Counter from collections

- [https://docs.python.org/3/library/collections.html#collections.Counter](https://docs.python.org/3/library/collections.html#collections.Counter)
Class `dict`

https://docs.python.org/3/library/stdtypes.html#typesmapping

**Sorting:**
- `sorted([1,4,3,2,8,9,6])`
  
  `[1, 2, 3, 4, 6, 8, 9]`

- `sorted([1,4,3,2,8,9,6], reverse=True)`
  
  `[9, 8, 6, 4, 3, 2, 1]`

- `list = sorted(dist.items(), key=lambda t: t[1], reverse=True)`

- `list[:10]`
  
Homework 4: Question 1

• Construct a WSJ text corpus that excludes both words tagged as –NONE- (see earlier example) and punctuation words (see previous slide)
• Show your Python console.
  i. How many words in the corpus?
  ii. How many distinct words?
  iii. Compute the lexical diversity (to 3 significant figures)
  iv. Plot the cumulative frequency distribution graph
  v. How many top frequency words do you need to account for at least 50% of the words in the corpus?
  vi. Print those words
Homework 4: Question 2

• Modify your text corpus in Question 1 to fold case. Recall:
  \[\text{text.count('the')}\] 50975
  \[\text{text.count('The')}\] 8442

• Recompute all the statistics required in Question 1.

• Did case folding make much difference?
Homework 4

• Usual rules
  • One PDF file!
  • Submit by next Wednesday (midnight)
  • We will review the homework in class next week this time