Corpus Words
counting word frequencies and processing text in Python with NLTK

Preliminary Text Processing Required :

- Define the words so that you can count them:
  - Filter out ‘junk data’
    - Formatting / extraneous material
    - First be sure it doesn’t reveal important information
  - Deal with upper / lower case issues
    - Ignore capitalization at beginning of sentence? Is “They” the same word as “they”?
    - Ignore other capitalization? In a name such as “Unilever Corporation” is “Corporation” the same word as “corporation”

Preliminary Text Processing Required (cont'd):

- Tokenization (or word segmentation):
  - Decide how to separate the characters in the sentence into individual words
    - Words are separated by "white space" or by special characters in English
    - No white space in Japanese language
    - In some languages, there are complex compound words – *"Lebensversicherungsgesellschaftsangestellter"*
  - Requires decisions on how to recognize and deal with punctuation
    - Apostrophes (one word *it's* vs. two words *it 's*
    - Hyphens (*snow-laden* vs. *New York-New Jersey*)
    - Periods (kept with abbreviations vs. separated as sentence markers)

Preliminary Processing Required: (cont'd)

- Morphology (To stem or not to stem?)
  - Depends on the application
  - With stemming
    - "cat" is the same word as "cats"
    - "computing" is the same word as "compute"
- Additional issues if OCR'd data or speech transcripts in order to correct transcription errors

Word Counting in Corpora

- Terminology for word occurrences:
  - Tokens – the total number of words
  - Distinct Tokens (sometimes called word types) – the number of distinct words, not counting repetitions

    - The following sentence from the Brown corpus has 16 tokens and 14 distinct tokens:
      They picnicked by the pool, then lay back on the grass and looked at the stars.

Word Frequencies

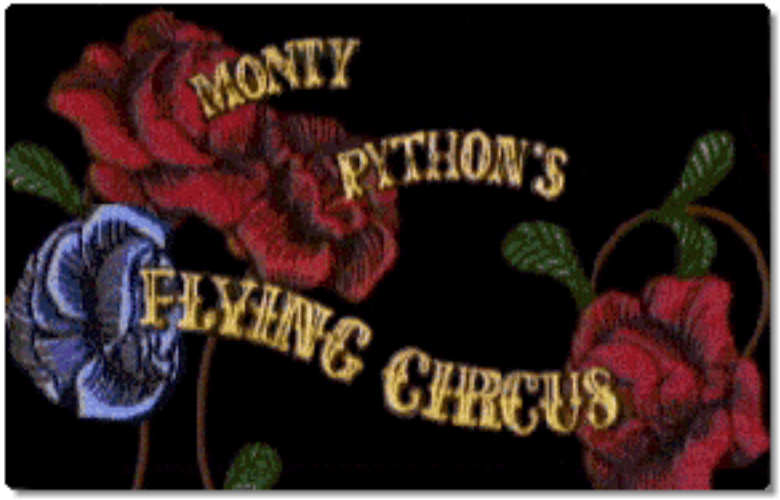
- Count the number of each token appearing in the corpus (or sometimes single document)
- A frequency distribution is a list of all tokens with their frequency, usually sorted in the order of decreasing frequency
- Used to make "word clouds"
  - For example, http://www.tumblr.com/tagged/word+cloud

Using NLTK in NLP

- NL ToolKit provides libraries of many of the common NLP processes at various language levels
  - Leverage these libraries to process text
- Goal is to learn about and understand how NLP can be used to process text without programming all processes
  - However, some programming is required to
    - Call libraries
    - Process data
    - Customize NLP processes
  - Programming language is Python

Python and NLP

- Python is freely available for many platforms from the Python Software Foundation:
  - http://www.python.org/
  - Named for the group Monty Python
  - We are using Python version 3.x
    - (not backward compatible with Python 2.x)



The group in 1969

Characteristics of Python

- Easy-to-learn scripting language, similar in many aspects to Perl
  - But with WYSIWYG block structure
- Object-oriented, with modules, classes, exceptions, high-level dynamic data types, similar to Java
- Strongly typed, but without type declarations (dynamic typing)
- Regular Expressions and other string processing features

- Many libraries offer wide functionality:
  - https://xkcd.com/353/

Natural Language Toolkit (NLTK)

- A suite of Python libraries for symbolic and statistical natural language programming
  - Developed at the University of Pennsylvania
- Developed to be a teaching tool and a platform for research NLP prototypes
  - Data types are packaged as classes
  - Goal of code is to be clear, rather than fastest performance
    - But increasingly production level software is made available through wrappers
- Latest version is compatible with Python 3.x
- Online book:
  http://www.nltk.org/book/
- Authors:
  Edward Loper, Ewan Kline
  and Steven Bird