

Assignment 3

Searching/Corpus linguistics

Due Monday, October 13, 2008

1. Someone has given you a file of all the students at Indiana and the classes they're taking this semester, and it's a comma-separated file, i.e., the different pieces of data are stored between commas. The general form, along with examples, is:

Course-Department, Course-Number, Name, ID, Major, Minor

LING,420,Herman Hermit,123456789,Linguistics,History

ECON,121,Dewey Finn,012345678,LINGUISTICS,Political Science

FREN,520,Polenta Polenta,91234567,French,linguistics

I want to find all linguistics majors (not minors) who are taking 400 level courses or higher this semester (irrespective of which department the 400-level course is in). Using `egrep`, write a regular expression to find what I'm looking for.

2. One notion sometimes talked about in linguistics is that of a **collocation**, two words which seem to have a meaning which is more than the sum of their parts. For example, *black box* is generally a collocation because it is not simply a box that is black; it's a box which you cannot see the contents of.
 - (a) Think of a good collocation in English; describe why the two words "go together"
 - (b) Go to the Corpus of Contemporary American English website (<http://www.americancorpus.org/>) and search for collocation. How many times does it appear? (e.g., *black box* appears 455 times)
 - (c) Simply knowing how often it appears doesn't say how good a collocation it is. Why not?
 - (d) How often does each word appear on its own? (e.g., *black* appears 145,546 times and *box* appears 36,817 times)
 - (e) Intuitively, do your numbers back up the idea that these two words "go together"?
 - (f) People use different tests to determine if a collocation is good, and one such test is *pointwise mutual information*—how informative each word is about the other.

$$(1) \text{pmi}(w_1, w_2) = \log_2 N \frac{C(w_1 w_2)}{C(w_1)C(w_2)}$$

- N is the total number of words (in this case, $N \approx 360,000,000$)

- $C(w_1w_2)$ is the count of the words next to each other ($C(\textit{black}, \textit{box}) = 455$)
- $C(w_1)$ ($= C(\textit{black}) = 145,546$)
- $C(w_2)$ ($= C(\textit{box}) = 36,817$)

In the case of *black box*, we obtain a score of 4.91. A score above 0 means that the words are at least somewhat related, so a score of 4.91 indicates a decent collocation. Negative numbers indicate that the words don't go together at all. Calculate the mutual information score for your two words and say why you think it got that score.

- Go to WordNet (<http://wordnet.princeton.edu/perl/webwn>) and look up the word *time*.
 - Focusing on the nouns, do you agree with the way *time* has been split into 10 meanings? Or, should some of them be put together? Pulled apart? Are there any meanings which are missing?
 - Go to Project Gutenberg (<http://www.gutenberg.org/>) and download a book in a text format (in English). Verify that there are at least 10 occurrences of the word *time* in this book. You can use multiple books if need be.
 - For the first 10 instances (or more, if you want) of *time*, assign them the appropriate meaning: $n1, \dots, n10$ for the nouns, $v1, \dots, v5$ for the verbs. In your write-up, include enough context so I can see whether I agree or not.
 - Take one of the meanings with multiple occurrences: is there a common pattern in the surrounding words? Think of a regular expression that would help you to identify this meaning in new text.
 - Bonus:** Use `egrep` on a new book, to see if your regular expression actually finds the meaning you want.
- You're given the following sentence, the first from *Ulysses* by James Joyce: *Stately, plump Buck Mulligan came from the stairhead, bearing a bowl of lather on which a mirror and a razor lay crossed.*
 - Write the words out vertically, one per line (column 1). For each word, assign it a lemma (column 2).
 - The following webpage defines the part-of-speech (POS) tags for the Penn Treebank: <http://www.comp.leeds.ac.uk/amalgam/tagsets/upenn.html> Using this tagset, assign POS tags to each word (column 3).
- Write a program that takes two strings and says what their shared prefix is, if any. e.g., *bring* and *behold* have a shared prefix *b*
 - Let's call the two strings `a` and `b` and assume that the first string is always shorter—i.e., `len(a) <= len(b)`
 - I (will) have some code available on the website to get you started.