Distant Reading and Text Mining of YikYak

"Dartmouth College in Fiction and in Fact"
Winter 2016
Dartmouth College
James.E.Dobson@Dartmouth.EDU

Connecting to Analysis System

- We need to connect from our own laptops to another system with the complete suite of tools.
- We will use an application called ‘ssh’ or the Secure Shell to get remote access to this system.
- All texts and applications will be on this remote system.

The “Dataset”

- Plain text dump of all Yaks from 5/22/2015 to 7/04/2015.
- Captures end of 15S, interim, and start of 15X
- Includes commencement day activities
- Complete dataset includes votes and timestamps.
- At this time you could have a “handle”—these are included in the complete dataset.

Examining Our Dataset

```
nltktool -f yaks-stripped.txt stats
opening file: yaks-stripped.txt
total number of lines: 3698
total number of words: 56019
total number of unique non-stop
words: 5750
```

topterms: Frequently Repeated Terms

- This function sorts the top twenty-five repetitions of a single term.
- No contextual information.
- Case-insensitive
- Removes many common words, such as articles (called “stopwords”).

```
nltktool -f yaks-stripped.txt
topterms
```

concordance: Keywords in Context (KWIC)

- Function displays the first twenty-five instances of keyword.
- Displays the keyword in its context.
- Useful for determining the variety of ways in which a term is used

```
nltktool -f yaks-stripped.txt
concordance —t love
```

search:
Locating the Missing Word

- Once you see some familiar patterns for word use, this tool can retrieve the ways in which it has been used.
- Pattern matching requires placing each word of phrase in angle brackets ‘<‘ and ‘>’.
- To locate the missing word in a phrase, add special search string (<.*>).

```
nltktool -f yaks-stripped.txt
search –t ‘<I> <love> (<.*>)’
```

search:
Locating Hashtags

- We can develop complex pattern searching phrases to find special terms, like a hashtag.

```
nltktool -f yaks-stripped.txt
search -t '<#> (<.*>)'
```

collocations: Finding Common Phrases

- Collocations are words that frequently appear next to each other.
- A list of common two word (bigram) phrases are returned by the program.

```
nltktool -f yaks-stripped.txt
collocations
```

similarity:
Words with a Similar Context

- The similarity function will show you words used in a similar manner as your search term.
- A list of similar words (but not the context) will be return.

```
nltktool -f yaks-stripped.txt
similarity -t 'kaf'
```

tf-idf
topic modeling

- Like “collocations” the tf-idf (term frequency-inverse document frequency) locates words or phrases that frequently occur together.
- We have two versions: single term and bigram (two word phrase).

```
nltktool -f yaks-
stripped.txt tf-idf

nltktool -f yaks-
stripped.txt tf-idf-ngram
```

$$\begin{aligned}\text{idf} &= -\log P(t|d) \\ &= \log \frac{1}{P(t|d)} \\ &= \log \frac{N}{|\{d \in D : t \in d\}|}\end{aligned}$$

Parts of Speech: Word Tagging

- The library used by nltktool also has the ability to tag parts of speech.
- This can help us identify possible references within the data.
- There are two functions in nltktool: locations and people.

```
nltktool -f yaks-stripped.txt
locations
```

Example: Top Locations

```
nltktool -f yaks-stripped.txt locations
opening file: yaks-stripped.txt
Dartmouth                                79
America                                  11
Greek                                    10
Hanover                                  10
New                                       8
Green                                     7
Novack                                    6
Boston                                    4
Foco                                      4
French                                    3
Long                                      3
Souleymane                                3
English                                   3
Japan                                     3
Collis                                    3
Kemeny                                    3
Canada                                    2
```

Advanced Tools

- Partitioning the dataset
  - Obtain just the most popular yaks
    - Determine threshold (100 votes)
    - Select votes greater than or equal to 100
    - Output into new file:

```
awk –F'|' '$2 >= 100 {print $1}'
yaks-sorted-votes.txt > pop-
yaks.txt
```

  - Run tools on new dataset:

```
nltktool –f pop-yaks.txt topterms
```