Introduction to Text Mining

School of Information Studies
Syracuse University

An Example of Unstructured Data

Friends and fellow citizens: I stand before you tonight under indictment for the alleged crime of having voted at the last presidential election, without having a lawful right to vote. It shall be my work this evening to prove to you that in thus voting, I not only committed no crime, but, instead, simply exercised my citizen's rights, guaranteed to me and all United States citizens by the National Constitution, beyond the power of any state to deny.

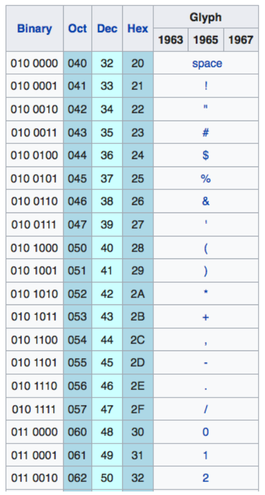
The preamble of the Federal Constitution says:

- "We, the people of the United States, in order to form a more perfect union, establish justice, insure domestic tranquility, provide for the common defense, promote the general welfare, and secure the blessings of liberty to ourselves and our posterity, do ordain and establish this Constitution for the United States of America."
- It was we, the people; not we, the white male citizens; nor yet we, the male citizens; but we, the whole people, who formed the Union. And we formed it, not to give the blessings of liberty, but to secure them; not to the half of ourselves and the half of our posterity, but to the whole people—women as well as men. And it is a downright mockery to talk to women of their enjoyment of the blessings of liberty while they are denied the use of the only means of securing them provided by this democratic-republican government—the ballot.

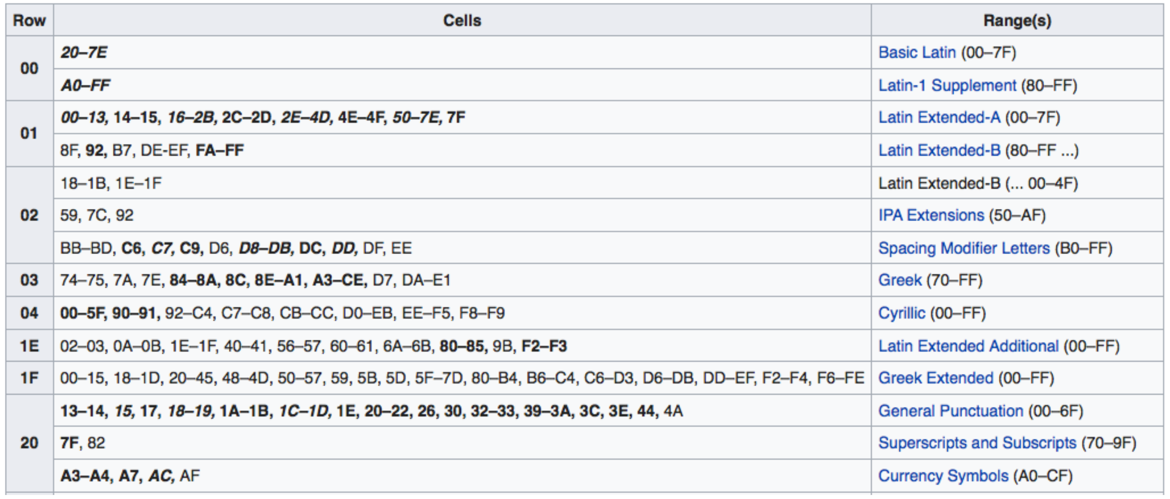
http://www.historyplace.com/speeches/anthony.htm

Inside the CPU, Text Is Easy

From a computer perspective—binary encoding for storing written language symbols from the world's various systems



Binary	Oct	Dec	Hex	Glyph		
				1963	1965	1967
010 0000	040	32	20		space	
010 0001	041	33	21		!	
010 0010	042	34	22		"	
010 0011	043	35	23		#	
010 0100	044	36	24		$	
010 0101	045	37	25		%	
010 0110	046	38	26		&	
010 0111	047	39	27		'	
010 1000	050	40	28		(	
010 1001	051	41	29		)	
010 1010	052	42	2A		*	
010 1011	053	43	2B		+	
010 1100	054	44	2C		,	
010 1101	055	45	2D		-	
010 1110	056	46	2E		.	
010 1111	057	47	2F		/	
011 0000	060	48	30		0	
011 0001	061	49	31		1	
011 0010	062	50	32		2	



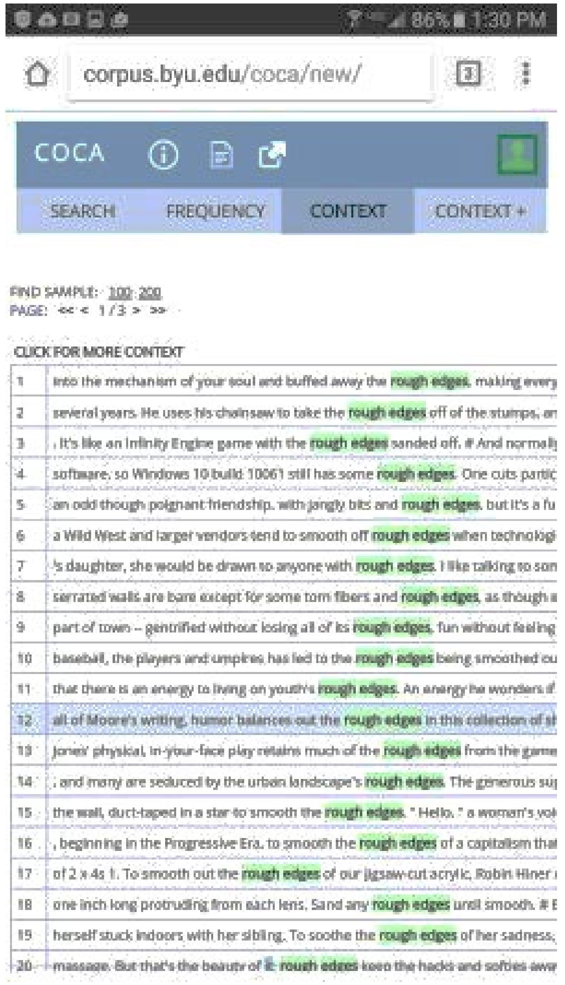
Row	Cells	Range(s)
00	***20–7E***	Basic Latin (00–7F)
	A0–FF	Latin-1 Supplement (80–FF)
01	***00–13*, 14–15, *16–2B*, 2C–2D, *2E–4D*, 4E–4F, *50–7E*, 7F**	Latin Extended-A (00–7F)
	8F, **92**, B7, DE-EF, **FA–FF**	Latin Extended-B (80–FF ...)
02	18–1B, 1E–1F	Latin Extended-B (... 00–4F)
	59, 7C, 92	IPA Extensions (50–AF)
	BB–BD, **C6**, ***C7***, **C9**, D6, ***D8–DB***, **DC**, ***DD***, DF, EE	Spacing Modifier Letters (B0–FF)
03	74–75, 7A, 7E, **84–8A**, **8C**, **8E–A1**, **A3–CE**, D7, DA–E1	Greek (70–FF)
04	**00–5F**, **90–91**, 92–C4, C7–C8, CB–CC, D0–EB, EE–F5, F8–F9	Cyrillic (00–FF)
1E	02–03, 0A–0B, 1E–1F, 40–41, 56–57, 60–61, 6A–6B, **80–85**, 9B, **F2–F3**	Latin Extended Additional (00–FF)
1F	00–15, 18–1D, 20–45, 48–4D, 50–57, 59, 5B, 5D, 5F–7D, 80–B4, B6–C4, C6–D3, D6–DB, DD–EF, F2–F4, F6–FE	Greek Extended (00–FF)
20	**13–14, *15*, 17, *18–19*, 1A–1B, *1C–1D*, 1E, 20–22, 26, 30, 32–33, 39–3A, 3C, 3E, 44,** 4A	General Punctuation (00–6F)
	7F, 82	Superscripts and Subscripts (70–9F)
	A3–A4, **A7**, ***AC***, AF	Currency Symbols (A0–CF)

From ASCII (7-bits) to Unicode: An international standard that supports up to four bytes (32 bits) per symbol and that encodes 137,439 characters from 146 different scripts
(Source: Wikipedia)

Outside the CPU, Text Is Difficult

From an analyst's perspective:

- Documents are organized into a corpus
- Each document contains an encoded sample of written language, generally:
  - Human readable files (e.g., plain text)
  - Metadata describing: Data source, date of capture, speaker/author, etc.
  - Character/symbol encoding appropriate to language
  - One file per document, or container file includes document separator characters (e.g., newline)
  - Each file contains unstructured, "natural" language content—generally the same "type" of content for each file in the corpus (i.e., don't mix tweets with books)



FIND SAMPLE: 100 200
PAGE: << < 1 / 3 > >>

CLICK FOR MORE CONTEXT

1	into the mechanism of your soul and buffed away the rough edges, making every
2	several years. He uses his chainsaw to take the rough edges off of the stumps, an
3	. It's like an Infinity Engine game with the rough edges sanded off. # And normally
4	software, so Windows 10 build 10061 still has some rough edges. One cuts partic
5	an odd though poignant friendship, with jangly bits and rough edges, but it's a fu
6	a Wild West and larger vendors tend to smooth off rough edges when technologi
7	's daughter, she would be drawn to anyone with rough edges. I like talking to son
8	serrated walls are bare except for some torn fibers and rough edges, as though a
9	part of town -- gentrified without losing all of its rough edges, fun without feeling
10	baseball, the players and umpires has led to the rough edges being smoothed ou
11	that there is an energy to living on youth's rough edges. An energy he wonders if
12	all of Moore's writing, humor balances out the rough edges in this collection of st
13	Jones' physical, in-your-face play retains much of the rough edges from the game
14	; and many are seduced by the urban landscape's rough edges. The generous sup
15	the wall, duct-taped in a star to smooth the rough edges. " Hello. " a woman's voi
16	, beginning in the Progressive Era, to smooth the rough edges of a capitalism that
17	of 2 x 4s 1. To smooth out the rough edges of our jigsaw-cut acrylic, Robin Hiner
18	one inch long protruding from each lens. Sand any rough edges until smooth. # E
19	herself stuck indoors with her sibling. To soothe the rough edges of her sadness,
20	massage. But that's the beauty of it: rough edges keep the hacks and softies awa

Text Analytics Overview

Text mining (TM):

- Looking for unexpected patterns in large data sets
- Focuses on statistical approaches such as counting word frequencies

Natural language processing (NLP):

- Understand how to program machines to make sense of human language
- Use linguistics to break text into its grammatical pieces, such as nouns and verbs

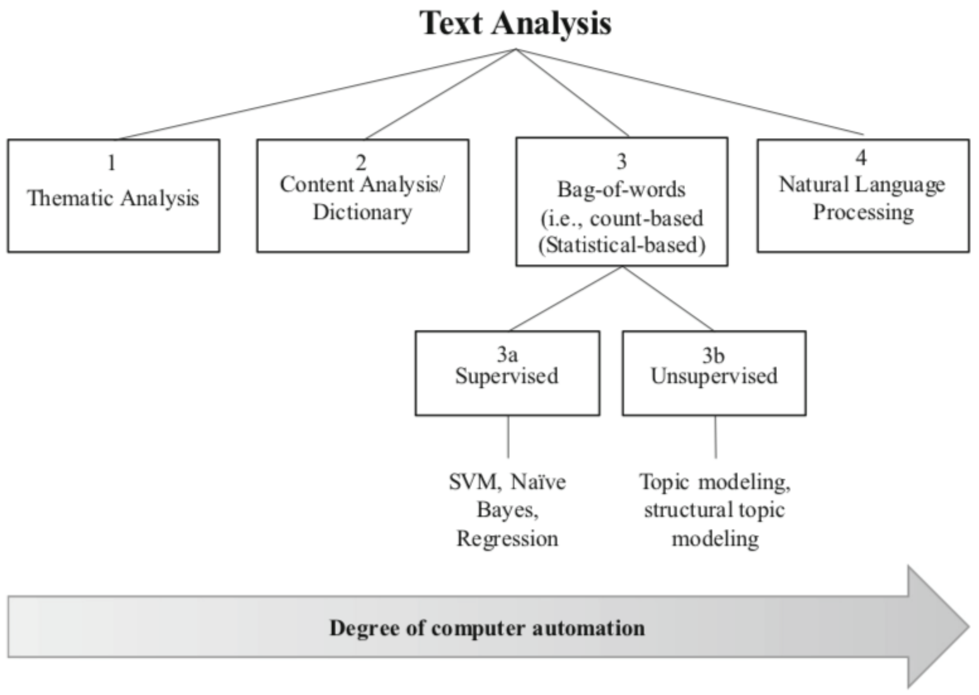

Text Analysis
1 Thematic Analysis
2 Content Analysis/ Dictionary
3 Bag-of-words (i.e., count-based (Statistical-based)
4 Natural Language Processing
3a Supervised
3b Unsupervised
SVM, Naïve Bayes, Regression
Topic modeling, structural topic modeling
Degree of computer automation

Image credit: Banks et al. (2018), Fig. 1, p. 447

“Unstructured” Is the Key

Natural language is notoriously flexible and ambiguous—even simple tasks like parsing are complicated.

- Contractions; compound words; misspellings; proper names; preposition attachment; vernacular

Transforming a set of documents into data that allow for comparisons, linkages, visualization, or other analysis, is **complex and contains many decisions.**

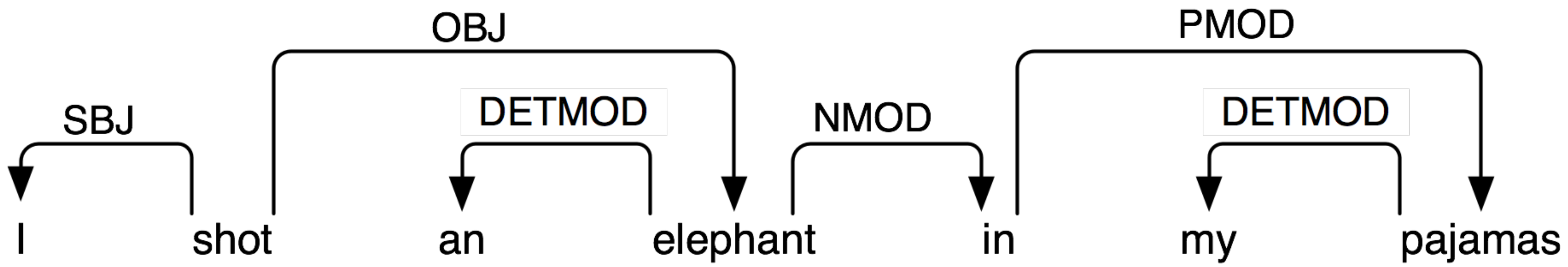
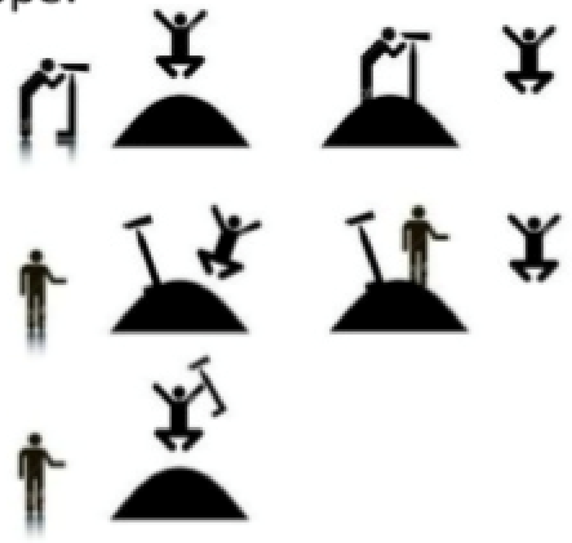


Image credit: NLTK.org

Example Challenge in Text Analysis: Preposition Attachment

I saw the man on the hill with a telescope.



1. I saw the man. The man was on the hill. I was using a telescope.
2. I saw the man. I was on the hill. I was using a telescope.
3. I saw the man. The man was on the hill. The hill had a telescope.
4. I saw the man. I was on the hill. The hill had a telescope.
5. I saw the man. The man was on the hill. I saw him using a telescope.

Image credit: Zareen Syed, UMBC

Question

If automated text analysis is so tricky, why bother? Why not just measure clicks and do surveys?

Introduction to Text Mining (cont.)

School of Information Studies
Syracuse University

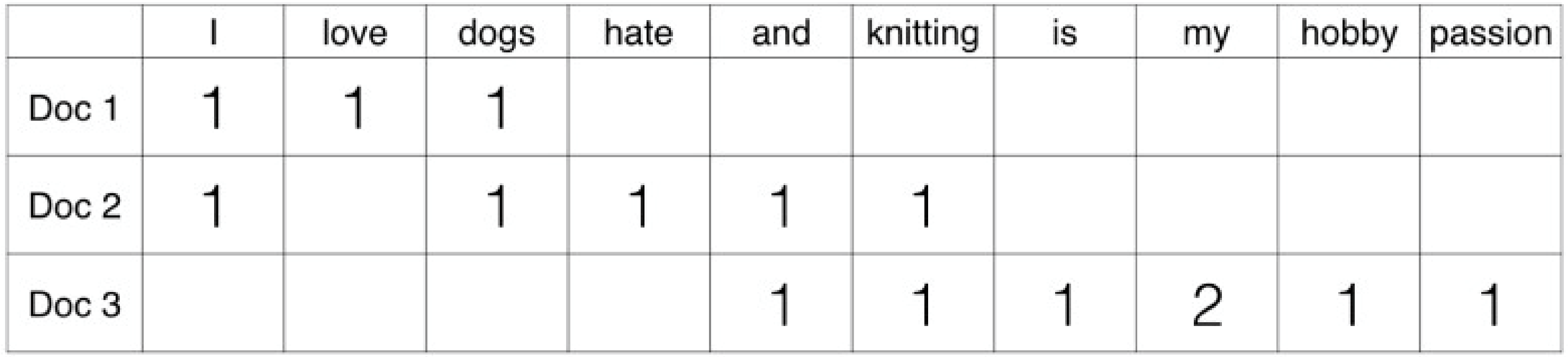
Question

If automated text analysis is so tricky, why bother? Why not just measure clicks and do surveys?

The Term-Document Matrix (AKA Document-Term Matrix)

School of Information Studies
Syracuse University

Statistical Approaches to Text

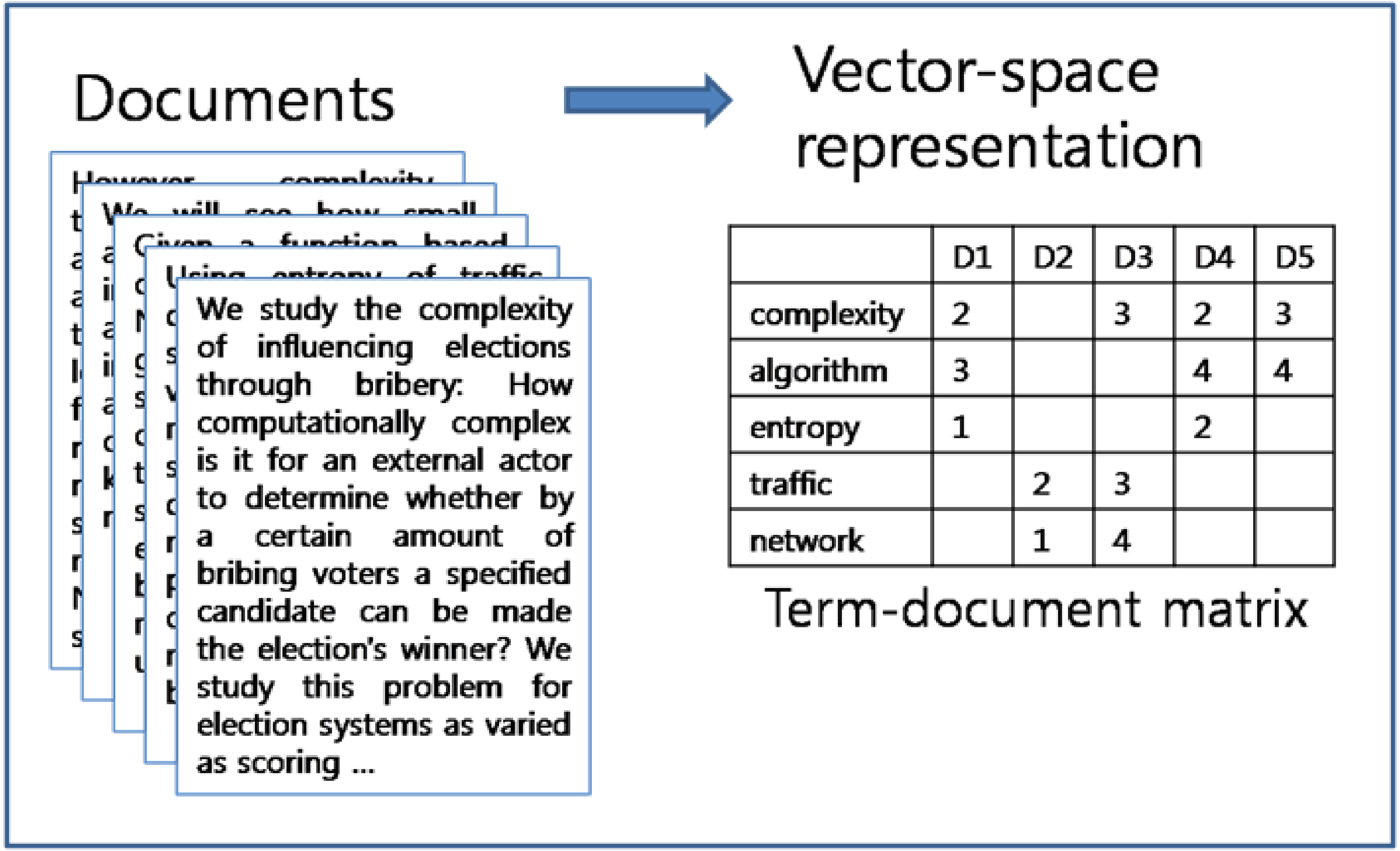


	I	love	dogs	hate	and	knitting	is	my	hobby	passion
Doc 1	1	1	1							
Doc 2	1		1	1	1	1				
Doc 3					1	1	1	2	1	1

Image credit: Pio Calrderon

- Treat each text fragment as a collection of units/words, without regard to word order
- Use many text fragments – as small as a phrase or as large as a book; each fragment is considered a “document”
- Compute a term-document matrix (TDM) or document-term matrix; a sparse matrix flagging the appearance of a term in a document
- Conduct statistical or machine learning analysis on the TDM to reveal patterns

Example TDM



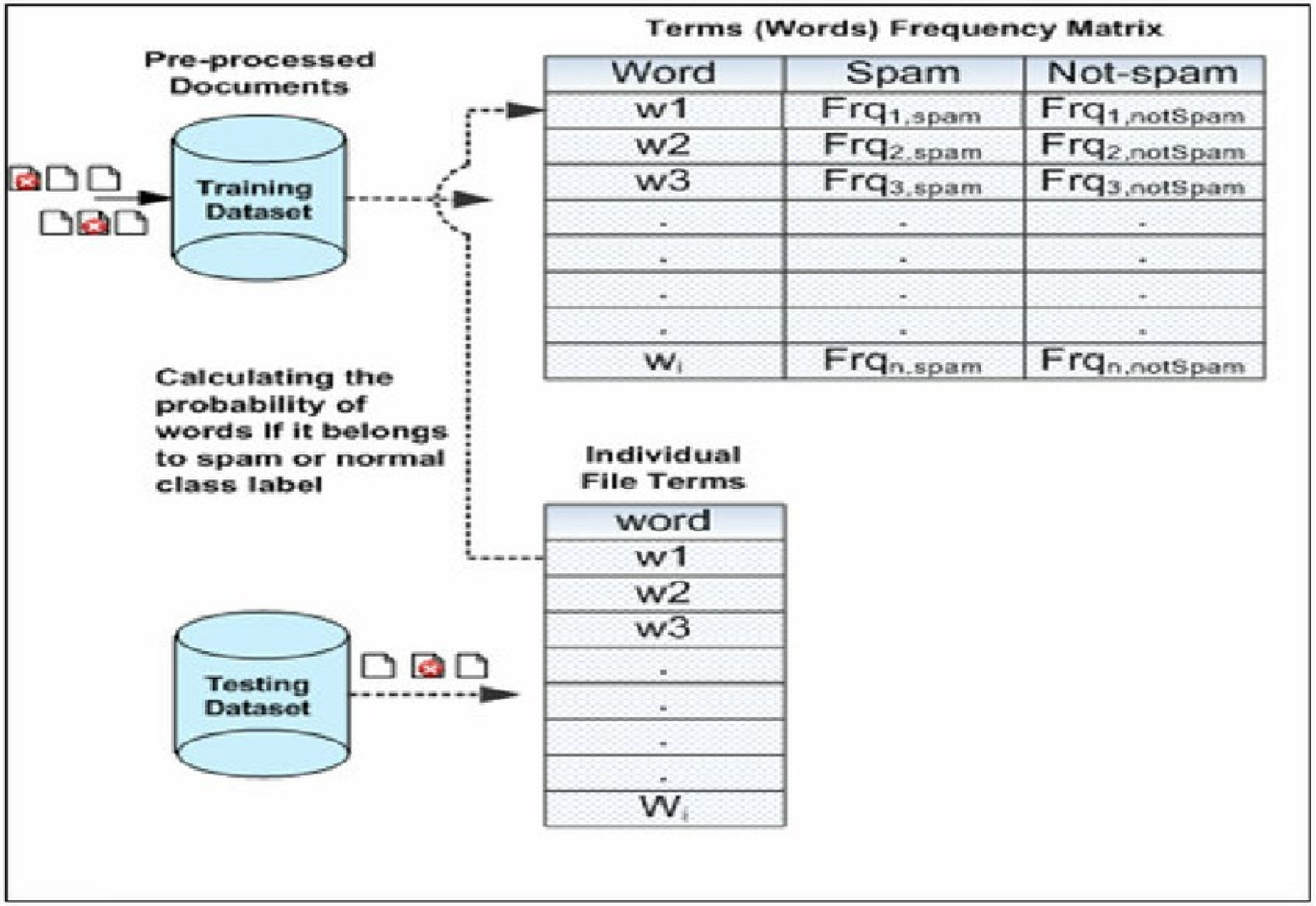
We study the complexity of influencing elections through bribery: How computationally complex is it for an external actor to determine whether by a certain amount of bribing voters a specified candidate can be made the election's winner? We study this problem for election systems as varied as scoring ...

	D1	D2	D3	D4	D5
complexity	2		3	2	3
algorithm	3			4	4
entropy	1			2	
traffic		2	3		
network		1	4		

Image Credit: SPE3DLab

School of Information Studies
Syracuse University

Example Application: Spam Filter


Pre-processed Documents
Training Dataset
Calculating the probability of words If it belongs to spam or normal class label
Testing Dataset
Individual File Terms

Terms (Words) Frequency Matrix

Word	Spam	Not-spam
w1	$Frq_{1,spam}$	$Frq_{1,notSpam}$
w2	$Frq_{2,spam}$	$Frq_{2,notSpam}$
w3	$Frq_{3,spam}$	$Frq_{3,notSpam}$
.	.	.
.	.	.
.	.	.
.	.	.
w_i	$Frq_{n,spam}$	$Frq_{n,notSpam}$

Individual File Terms

word
w1
w2
w3
.
.
.
.
W_i

Image Credit: Hassan & Hmeidi, 2008

Question

Describe a useful application of a term-document matrix?

Creating Word Clouds

School of Information Studies
Syracuse University

Read Some Web Text

```
url <- "http://programdevelopment.syr.edu/wp-content/uploads/2018/03/REM_Manuscript_fall2017_text.txt"
charVector <- scan(url, character(0), sep = "\n")
head(charVector)
```

```
'SYRACUSE MANUSCRIPT'
 'FALL 2017 | VOL. 6 | NO. 2'
 'SYRACUSE UNIVERSITY\'S AFRICAN AMERICAN AND LATINO ALUMNI MAGAZINE'
 'ON THE COVER:'
'Left to right, from top: Cheryl Wills \'89 and Taye Diggs \'93; Lazarus Sims
\'96; Lt. Col. Pia W. Rogers \'98, G\'01, L\'01 and Dr. Akima H. Rogers \'94;'
'Amber Hunter \'19, Nerys Castillo-Santana \'19, and Nordia Mullings \'19; Demaris
Mercado \'92; Dr. Ruth Chen and Chancellor Kent Syverud; Carmelo Anthony; Darlene
Harris \'84 and Debbie Harris \'84 with Soledad O\'Brien'
```

Library(quanteda)

The quanteda package was developed by Kenneth Benoit (Kenbenoit) at the London School of Economics.

Does everything that TM does (an older package), plus much more:

- Includes readtext(), a wrapper for a wide range of text files (PDF, CSV, text, XML, JSON) that simplifies reading in folders of text files
- Directly supports document-level variables for machine learning
- A comprehensive text processing package that uses some C++ and Fortran—but unlike Stanford Core NLP tools, no Java; mostly runs faster and with less memory than Stanford Core NLP

Citation: Benoit, Kenneth, Kohei Watanabe, Haiyan Wang, Paul Nulty, Adam Obeng, Stefan Müller, and Akitaka Matsuo. (2018) "quanteda: An R package for the quantitative analysis of textual data". *Journal of Open Source Software.* 3(30), 774. https://doi.org/10.21105/joss.00774

Text Transformations

Stemming (removing the endings of words)

Removing the punctuation

Removing low frequency terms

Taking out the “stop” words

Words such as “*the*,” “*a*,” and “*at*” appear in so many different parts of the text that they are useless for differentiating between documents.

Additional possible transformations not shown here: lowercase, removing numbers

Preparing and Visualizing Corpus

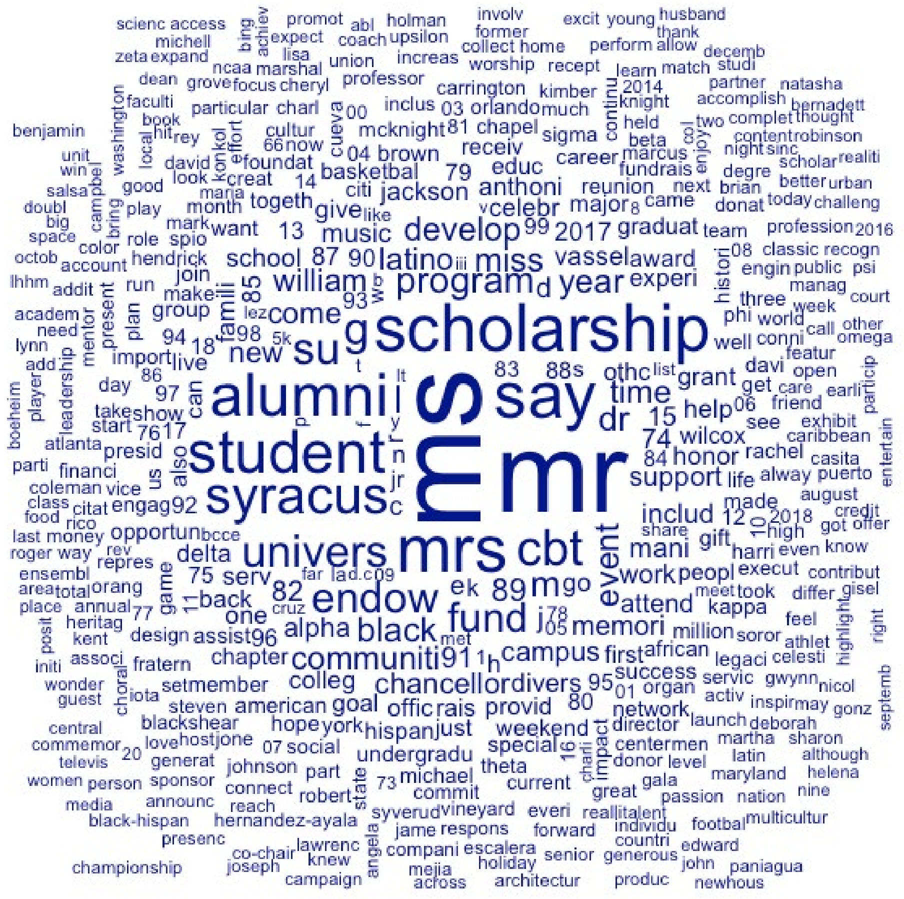
```
#Make a corpus of web data
intcorpus <- corpus(charVector)
paras <- corpus_reshape(intcorpus, to="paragraphs")

#Turn the corpus into a document-term matrix while removing stop words and punctuation
webfile_dtm <- dfm(paras, stem=TRUE, remove_punct=TRUE,  remove=stopwords("english"))
webfile_dtm <- dfm_trim(webfile_dtm, min_termfreq=2)

#2,079 documents, 976 features (99.6% sparse)
webfile_dtm

#Visualize words
textplot_wordcloud(webfile_dtm)
```

textplot_wordcloud(webfile_dtm)



School of Information Studies
Syracuse University

Let’s Trim Out Some Useless Words

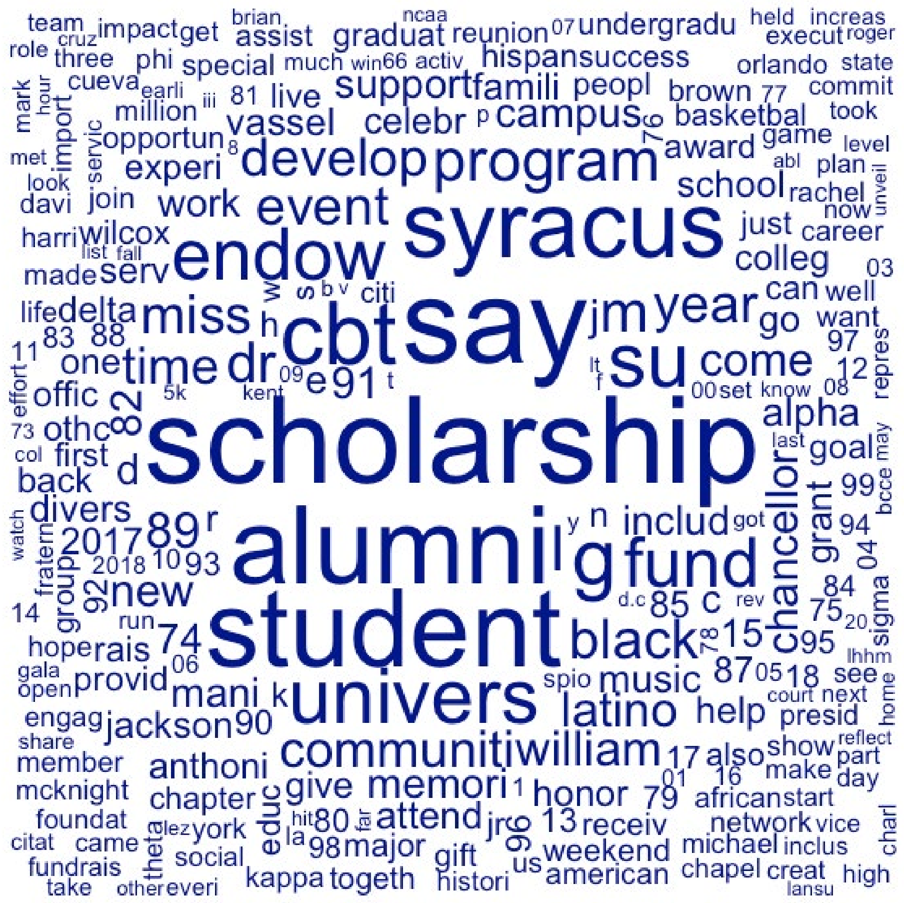
```
webfile_dtm <- dfm(paras, stem=TRUE, remove_punct=TRUE,
                                        remove=c(stopwords("english"),"ms","mr","mrs"))
webfile_dtm <- dfm_trim(webfile_dtm, min_termfreq=3)

#2,079 documents, 973 features (99.6% sparse)
webfile_dtm

textplot_wordcloud(webfile_dtm)
```

Three fewer features than before because we removed the nuisance words in addition to standard stop words.

textplot_wordcloud(webfile_dtm)



School of Information Studies
Syracuse University

Let’s Raise the Bar

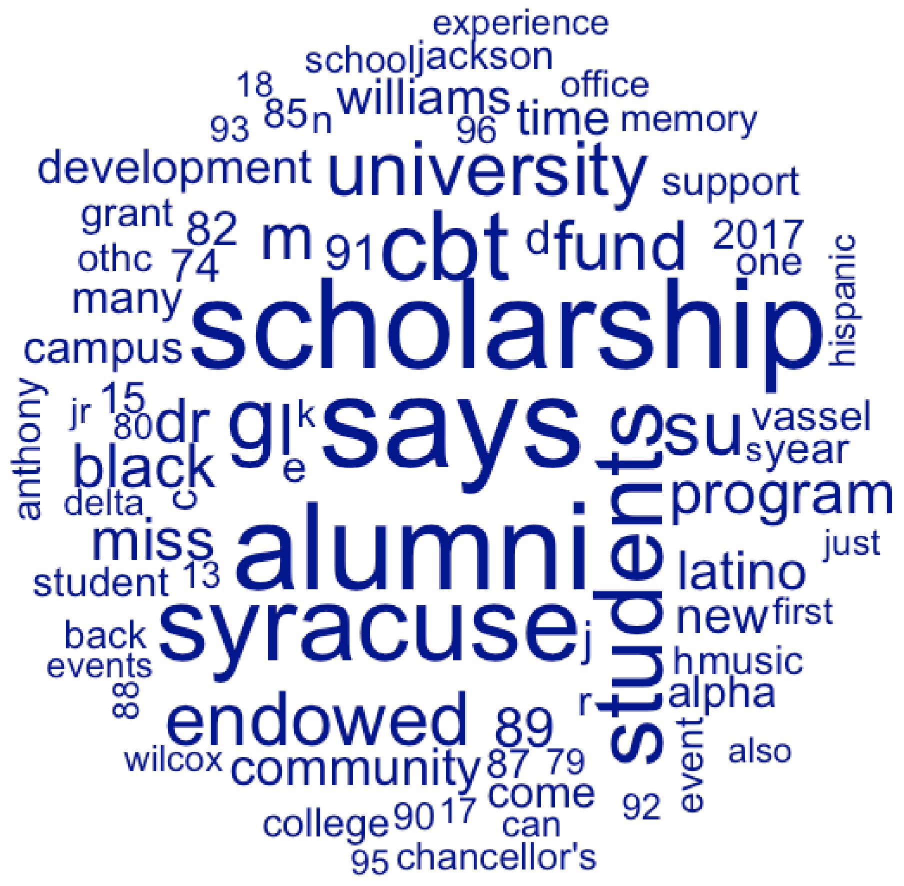
```
#Raise the frequency threshold and turn off stemming
webfile_dtm <- dfm(paras, stem=FALSE, remove_punct=TRUE,
                        remove=c(stopwords("english"),"ms","mr","mrs"))

#Set a minimum word frequency
webfile_dtm <- dfm_trim(webfile_dtm, min_termfreq=18)

textplot_wordcloud(webfile_dtm)
```

By raising min_termfreq, we eliminate words from the TDM that are less frequent. This can be dangerous for indexing! Infrequent words can be powerful!

textplot_wordcloud(webfile_dtm)



School of Information Studies
Syracuse University

Question

Justify when it might be appropriate to use a word cloud as part of a text analysis.

Creating Word Clouds (cont.)

School of Information Studies
Syracuse University

Answer

Justify when it might be appropriate to use a word cloud as part of a text analysis.

- Example: get a “feeling” for answers from a survey response

Sentiment Analysis

School of Information Studies
Syracuse University

Conceptual Methodology

Load positive and negative word lists

Count positive words and negative words that match those in the word list from the TDM

Compute ratios: positive/total and negative/total

```
#Reprocess the TDM to include low frequency words
webfile_dtm <- dfm(paras, stem=FALSE, remove_punct=TRUE,
              remove=c(stopwords("english"),"ms","mr","mrs"))
webfile_dtm <- dfm_trim(webfile_dtm, min_termfreq=1)

#2,079 documents, 3,735 features (99.9% sparse)
webfile_dtm
```

1. Make a “Named Number” List of Words

```
#Create a simple matrix from the TDM
webWords <- as.matrix(webfile_dtm)
str(webWords)
```

All we have done is simplify the more complex—and more efficiently stored TDM—into a regular rectangular matrix.

```
num [1:2079, 1:3735] 1 0 1 0 0 0 0 0 0 0 ...
 - attr(*, "dimnames")=List of 2
  ..$ docs    : chr [1:2079] "text1.1" "text2.1" "text3.1" "text4.1" ...
  ..$ features: chr [1:3735] "syracuse" "manuscript" "fall" "2017" ...
```

Note that the columns of this matrix are labeled with the terms from the TDM.

1. Make a “Named Number” List of Words (cont.)

```
#Sum the columns to get the overall term counts
wordCounts <- colSums(webWords)
wordCounts <- sort(wordCounts, decreasing=TRUE)

#Check the first several items in "wordCounts"
head(wordCounts, 20)
```

```
says    scholarship        alumni       syracuse      cbt       students
107        104               100           85          77          76
 g         su           university      endowed        1          fund
70         58                57            56          51          50
 m         dr              black        program      miss         89
47         44                43            42          42          37
```

These are “named” numbers: a numeric vector but with a text attribute that stores a label for each value

School of Information Studies
Syracuse University

2. Read the Positive and Negative Dictionaries

```
#2006 items
posWords <- scan("positive-words.txt",   character(0), sep = "\n")

#4783 items
negWords <- scan("negative-words.txt",   character(0), sep = "\n")

head(posWords,18)
```

```
 [1]  "a+"             "abound"          "abounds"      "abundance"
 [5]  "abundant"       "accessable"      "accessible"   "acclaim"
 [9]  "acclaimed"      "acclamation"     "accolade"     "accolades"
[13]  "accommodative"  "accomodative"    "accomplish"   "accomplished"
[17]  "accomplishment" "accomplishments"
```

3. Match Positive and Negative Words

matchedP <- match(names(wordCounts), posWords, nomatch = 0)

matchedN <- match(names(wordCounts), negWords, nomatch = 0)

- nomatch=0 specifies what value ("0") to use when an item has no match in the second vector

Here are the first 100 values of matchedP:

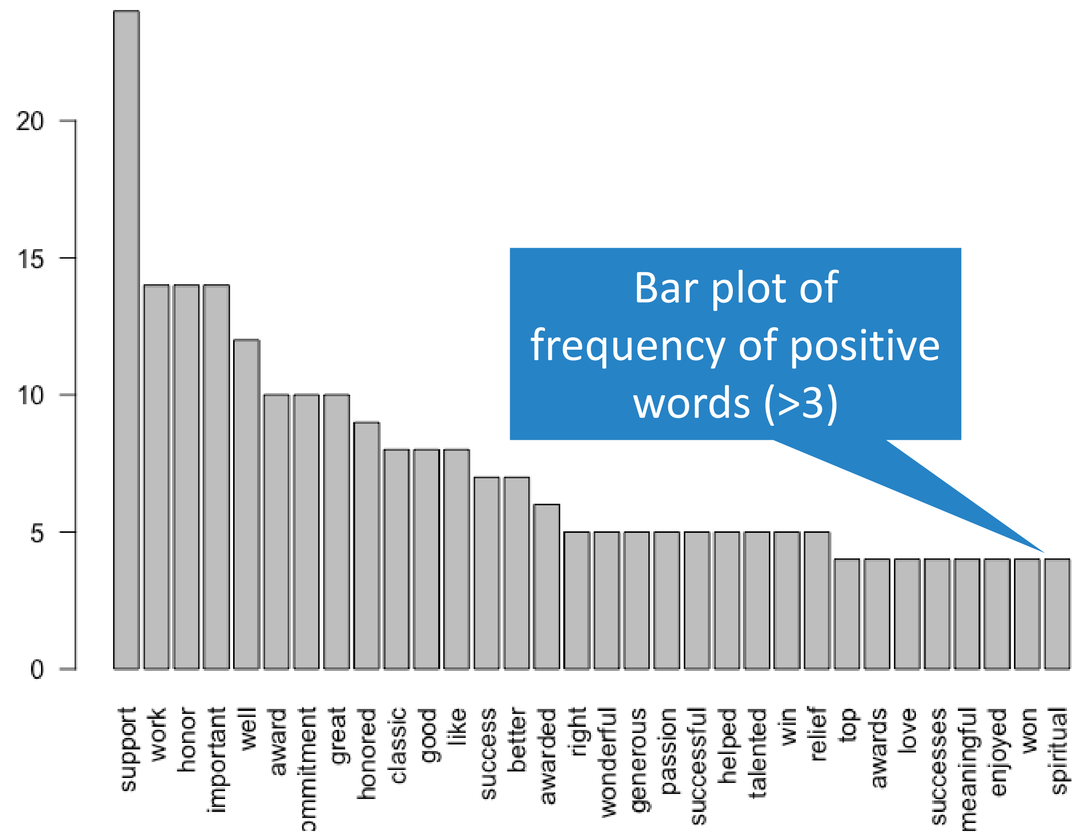
```
head(matchedP,100)
  [1]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
 [16]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
 [31]    0    0    0    0    0    0    0    0    0    0 1746    0    0    0    0
 [46]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
 [61]    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
 [76]    0    0    0    0    0    0    0    0    0    0    0         0    0    0
 [91]    0    0    0    0    0    0    0    0    0    0
Names (wordCounts) [41]
[1]  "support"
posWords[1746]
[1]  "support"
```

Item 41 in names(wordCounts) matches with item 1746 in posWords.

Matched?

matchedP is mostly empty space—lots of zeros.

Where matchedP is not empty, it just contains an index reference to posWords.



Bar plot of frequency of positive words (>3)

4. Calculate the Proportions!

```
#Calculate the proportion of words that are positive or negative
sum(matchedP)/totalWords
[1] 0.04228368

sum(matchedN)/totalWords
[1] 0.01248885
```

This document contains about 3.4 times as many positive words as negative words.

- Is this a surprising result? Why or why not?

Questions

Does this truly measure sentiment? Where could it go wrong?

What are some circumstances where we might draw the wrong conclusions?

Sentiment Analysis (cont.)

School of Information Studies
Syracuse University

Answers

Does this truly measure sentiment? Where could it go wrong?

What are some circumstances where we might draw the wrong conclusions?

How about:

- “No, that movie was not bad.”
- “I liked the movie. The lead was a terrible person, and his partner was even worse.”