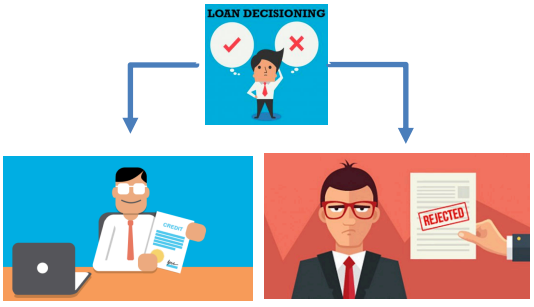
Data Science in the real world

School of Information Studies
SYRACUSE UNIVERSITY
ischool.syr.edu

Credit Card Decisioning

Credit Card Decisioning



Credit Card Decisioning

Credit Decisioning

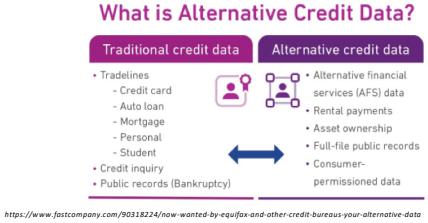
→What data / attributes should be used?

→ What is possible?

Credit Decisioning

→What data / attributes should be used?

→ What is possible?

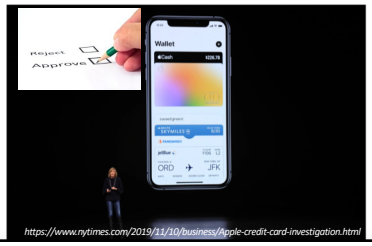


Apple's New Credit Card

The New York Times

Apple Card Investigated After Gender Discrimination Complaints

A prominent software developer said on Twitter that the credit card was "sexist" against women applying for credit.



https://www.nytimes.com/2019/11/10/business/Apple-credit-card-investigation.html

NY Times Article

- **David Hansson (Husband) noted :**
  - His spouse, Jamie Hansson, had a better credit score and other factors in her favor
  - Her application for a credit line increase had been denied.
- **Mr. Hansson, a prominent software developer, wondered**
  - how his credit line could be 20 times higher,
  - referring to Apple Card as a "sexist program"
- *The card is a partnership between Apple and Goldman Sachs*

The Potential of Machine Learning Bias

Steve Wozniak
@stevewoz

Replying to @dhh @AppleCard

The same thing happened to us. I got 10x the credit limit. We have no separate bank or credit card accounts or any separate assets. Hard to get to a human for a correction though. It's big tech in 2019.

♡ 3,801 6:51 PM - Nov 9, 2019

The Potential of Machine Learning Bias

Machine Learning Bias: How to Know if there is bias?

→**New York State regulators** announced that they would investigate the algorithm used by Apple Card to determine the creditworthiness of applicants

→"Any algorithm that intentionally or not results in discriminatory treatment of women or any other protected class **violates New York law**"

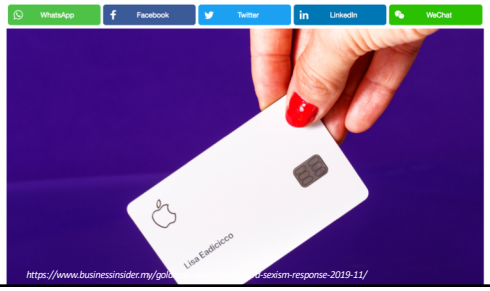
Credit Decisioning

→Do you think that apple used the "gender" attribute?

Goldman Sachs will let people appeal their Apple Card credit limit after allegations of sexist algorithms

Isobel Asher Hamilton, Business Insider US

November 12, 2019



Text Mining

Professor Jeff Saltz

Professor Jeff Stanton

School of Information Studies
SYRACUSE UNIVERSITY

ischool.syr.edu

Summary of Previous Learning

What you should know and be able to do at this point :

1. List major skills needed by data scientists and describe the development of a DS project with domain analysis, SMEs, data and modeling
2. Use data frames in R as well as more complex data structures; use multiple strategies for accessing external data from R; use SQL facilities from within R; automate with functions
3. Use and interpret the most common descriptive statistics; describe the effects of randomness on sampling; create and interpret a sampling distribution
4. Use plot and ggplot to visualize data and create maps
5. Create and interpret a multiple regression model using lm()
6. Run a "market basket" analysis using arules; coerce a set of factor variables into a transaction matrix
7. Define, run, and interpret a "supervised" data mining model such as a classification/regression tree (CART) or support vector machine (SVM)

Objectives for this Week

- Gain experience analyzing **unstructured data**
- Define and describe the application of **text mining** techniques such as:
  - **Word clouds**
  - **Sentiment analysis**
- Practice text mining techniques in R
- Data Science in the real world

Introduction to Text Mining

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:

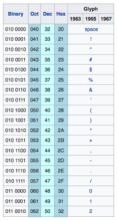
"We, the people of the United States, in order to form a more perfect union, establish justice, insure domestic tranquility, provide for the common defense, promote the general welfare, and secure the blessings of liberty to ourselves and our posterity, do ordain and establish this Constitution for the United States of America."

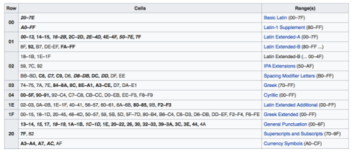
It was we, the people; not we, the white male citizens; nor yet we, the male citizens; but we, the whole people, who formed the Union. And we formed it, not to give the blessings of liberty, but to secure them; not to the half of ourselves and the half of our posterity, but to the whole people—women 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





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)

"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**

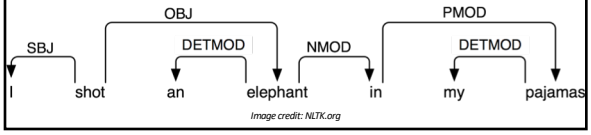


Image credit: NLTK.org

Example Challenge in Text Analysis

I saw the man on the hill with a telescope

Example Challenge in Text Analysis

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

Example Challenge in Text Analysis

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

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

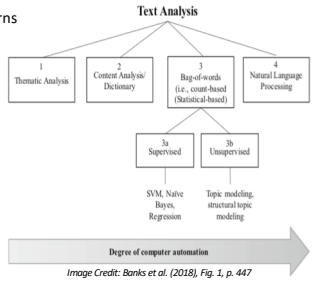


Image Credit: Banks et al. (2018), Fig. 1, p. 447

Question

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

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

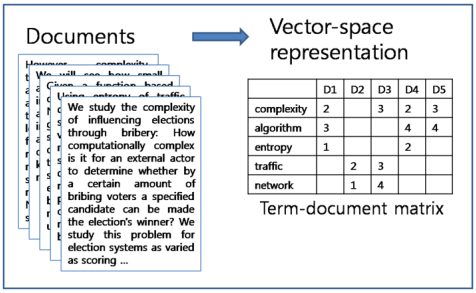
Statistical Approaches to Text

	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

- 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

Image credit: Pio Calrderon

Example TDM



	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.

Example Application: Spam Filter

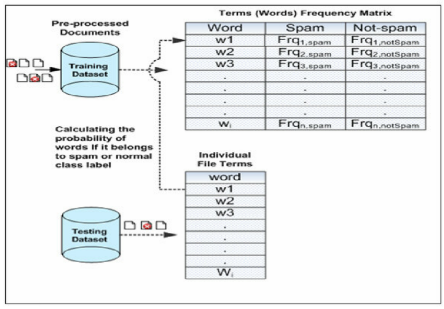


Image Credit: Hassan & Hmeidi, 2008

Question

- Describe another useful application of a term-document matrix?

Worked Example: Word Cloud

Read Some Web Text

```
url <- "https://intro-datascience.s3.us-east-2.amazonaws.com/MLK-dream.txt"

charVector <- scan(url, character(0), sep = "\n")

head(charVector)
```

[1] "I am happy to join with you today in what will go down in history as the greatest demonstration for freedom in the history of our nation."

library(quanteda)

- The quanteda package was developed by Kenneth Benoit (http://kenbenoit.net) at the London School of Economics
- Does everything that tm does (in book), 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

```
#install.packages("quanteda")
#install.packages("quanteda.textstats")
#install.packages("quanteda.textplots")
library(quanteda)
library(quanteda.textstats)
library(quanteda.textplots)

# Make a corpus of web data
corp <- corpus(charVector)

#remove punctuation
toks <- tokens(corp, remove_punct=TRUE)
```

Preparing and Visualizing Corpus

```
#remove stopwords
toks_nostop <- tokens_select(toks, pattern = stopwords("en"),
                                          selection = "remove")

#create a dfm
dfm <- dfm(toks_nostop)

#show most frequent words
freqWords <- textstat_frequency(dfm)
freqWords[1:5, c("feature", "frequency")]
 feature frequency
1         38
2   negro      14
3     one      13
4  nation      11
5     day      11
```

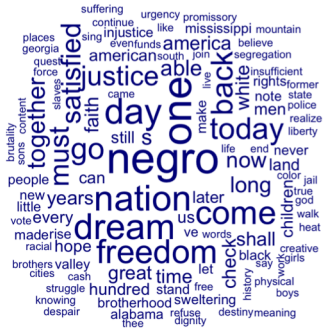
Preparing and Visualizing Corpus

```
#remove stopwords
toks_nostop <- tokens_select(toks, pattern = c(stopwords("english"),""),
                                                  selection = "remove")

#create a dfm
dfm <- dfm(toks_nostop)

#show most frequent words
freqWords <- textstat_frequency(dfm)
freqWords[1:4, c("feature", "frequency")]
 feature frequency
1   negro      14
2     one      13
3  nation      11
4     day      11
```

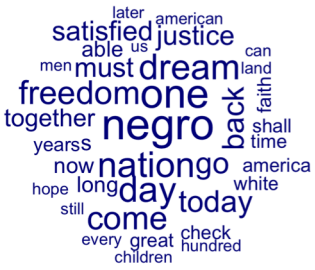
Preparing and Visualizing Corpus

```
#remove stopwords
toks_nostop <- tokens_select(toks, pattern = c(stopwords("english"),""),
                                                  selection = "remove")

#create a dfm
dfm <- dfm(toks_nostop)

#show most frequent words
freqWords <- textstat_frequency(dfm)
freqWords[1:4, c("feature", "frequency")]
 feature frequency
1   negro      14
2     one      13
3  nation      11
4     day      11
```

textplot_wordcloud(dfm, min_count=2)



textplot_wordcloud(dfm, min_count=4)



Question

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

Sentiment Analysis

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

```
# First count total words in file
totalWords <- sum(freqWords$frequency)
totalWords
[1] 738
```

1. Read the Pos & Neg Dictionaries

```
#positive and negative word file locations
posFile <- "https://intro-datascience.s3.us-east-2.amazonaws.com/positive-words.txt"
negFile <- "https://intro-datascience.s3.us-east-2.amazonaws.com/negative-words.txt"

#read in positive and negative word files
posWords <- scan(posFile,   character(0), sep = "\n")
negWords <- scan(negFile,   character(0), sep = "\n")

#remove header info
posWords <- posWords[-1:-34]
negWords <- negWords[-1:-34]

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"
```

2. Match & Count the Positive Words

```
#match the words
posDFM <- dfm_match(dfm, posWords)

#count the words
freqWords <- textstat_frequency(posDFM)
freqWords[1:5, c("feature", "frequency")]
 feature frequency
1    freedom        10
2  satisfied         8
3      faith         5
4      great         5
5   creative         2

#total num positive words
totalPos <- sum(freqWords$frequency)
totalPos
[1] 78
```

2. Match & Count the Negative Words

```
#match the words
negDFM <- dfm_match(dfm, negWords)

#count the words
freqWords <- textstat_frequency(negDFM)
freqWords[1:5, c("feature", "frequency")]
 feature frequency
feature frequency
1     injustice         3
2     brutality         2
3       despair         2
4  insufficient         2
5        refuse         2

totalNeg <- sum(freqWords$frequency)
totalNeg
[1] 63
```

4. Calculate the Proportions!

```
#calculate the prop of words that are pos or neg
totalPos - totalNeg
[1] 15

totalPos / totalWords
0.1056911

totalNeg / totalWords
0.08536585
```

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

Questions

- What does this process – dictionary matching and term frequency counting – measure?
- What are some circumstances where we might draw the wrong conclusions?

This Week in Lab

- You will read in a "corpus" (a text file with multiple sections), along with the positive and negative word dictionaries, and make a TDM
- The TDM is useful for a wide variety of statistical and machine learning analyses, but we will use it in a simple way – to get total counts of term frequencies
- Matching terms between the corpus and the dictionaries will provide opportunities for visualization and summary statistics (e.g., ratio of positive to negative words)
- Last Homework!