Principles of Text Analysis

Patrick van Kessel, Senior Data Scientist
Pew Research Center

Agenda

- Basic principles: how to convert text into quantitative data
- Overview of common methods: a map of useful analysis tools
- Demo: text analysis in action

The role of text in social research

The role of text in social research

Why text?

- Free of assumptions
- Potential for richer insights relative to closed-format responses
- If organic, then data collection costs are often negligible

The role of text in social research
Where do I find it?

- Open-ended surveys / focus groups / transcripts / interviews
- Social media data (tweets, FB posts, etc.)
- Long-form content (articles, notes, logs, etc.)

The role of text in social research
What makes it challenging?

- Messy
  - “Data spaghetti” with little or no structure
- Sparse
  - Low information-to-data ratio (lots of hay, few needles)
- Often organic (rather than designed)
  - Can be naturally generated by people and processes
  - Often without a research use in mind

Data selection and preparation

Data selection and preparation

- Know your objective and subject matter (if needed find subject matter expert)
- Get familiar with the data
- Don’t make assumptions - **know your data, quirks and all**

Data selection and preparation
Text Acquisition and Preprepation

Select relevant data (text corpus)

- Content
- Metadata

Prepare the input file

- Determine unit of analysis
- Process text to get one document per unit of analysis

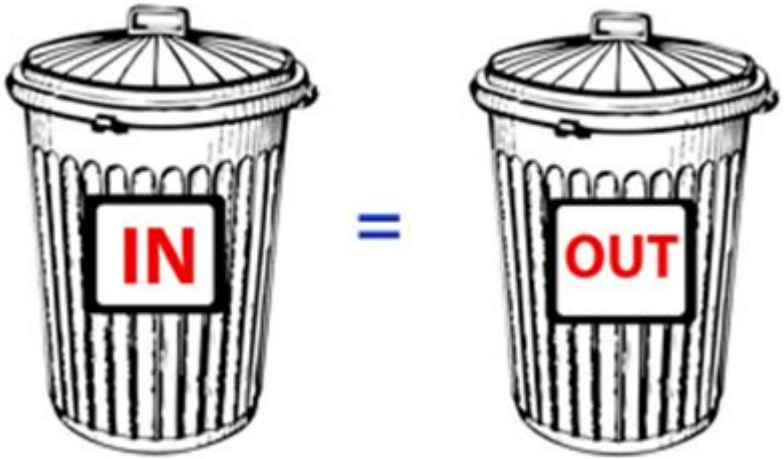


Image credit: http://www.nickmilton.com/2016/12/garbage-lessons-in-garbage-knowledge-out.html

(Pre-)Processing
Turning text into data

Turning text into data

- How do we sift through text and produce insight?
- Might first try searching for keywords
- How many times is “analysis” mentioned?

	Raw Documents	
1	Text **analysis** is fun	
2	I enjoy analyzing text data	
3	Data science often involves text analytics	

Turning text into data

- How do we sift through text and produce insight?
- Might first try searching for keywords
- How many times is “analysis” mentioned?

We missed this one

	Raw Documents	
1	Text **analysis** is fu	
2	I enjoy analyzing text data	
3	Data science often involves text analytics	

And this one too

Turning text into data

- Variations of words can have the same meaning but look completely different to a computer

	Raw Documents	
1	Text **analysis** is fun	
2	I enjoy analyzing text data	
3	Data science often involves text analytics	

Turning text into data
Regular Expressions

- A more sophisticated solution: regular expressions
- Syntax for defining string (text) patterns

	Raw Documents	
1	Text analysis is fun	
2	I enjoy analyzing text data	
3	Data science often involves text analytics	

Turning text into data
Regular Expressions

- Can use to search text or extract specific chunks
- Example use cases:
  - Extracting dates
  - Finding URLs
  - Identifying names/entities
- https://regex101.com/
- http://www.regexlib.com/

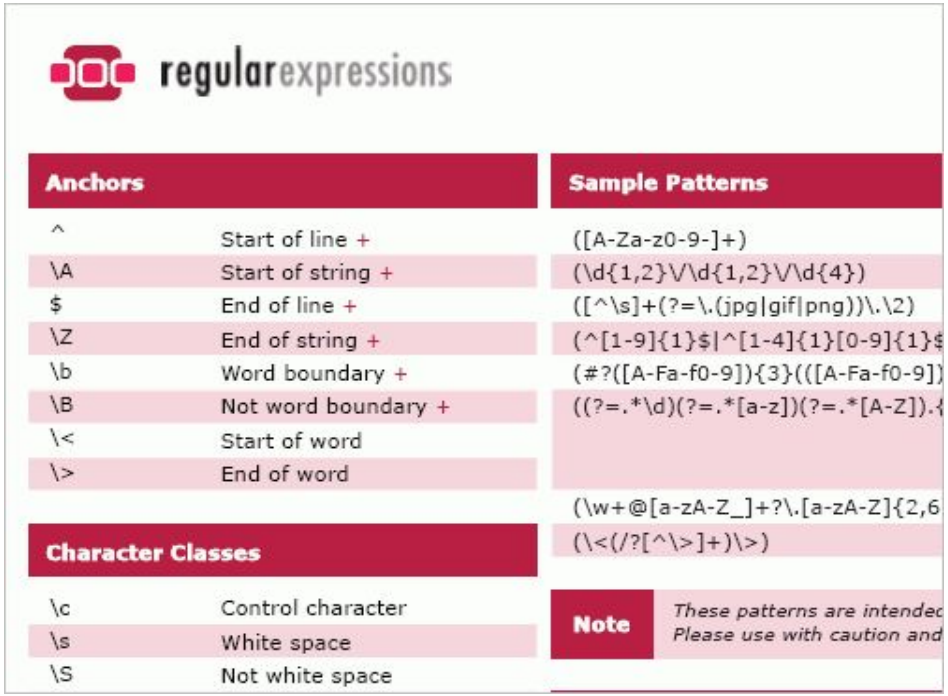


Image credit: https://www.smashingmagazine.com/2009/06/essential-guide-to-regular-expressions-tools-tutorials-and-resources/

Turning text into data
Regular Expressions

\banaly[a-z]+\b

	Raw Documents	
1	Text **analysis** is fun	
2	I enjoy **analyzing** text data	
3	Data science often involves text **analytics**	

Turning text into data
Regular Expressions

Regular expressions can be extremely powerful…

...and terrifyingly complex:

URLS: ((https?:\/\/(www\.)?)?[-a-zA-Z0-9@:%._\+~#=]{2,4096}\.[a-z]{2,6}\b([-a-zA-Z0-9@:%_\+.~#?&//=]*))

DOMAINS: (?:http[s]?\:\/\/)?(?:www(?:s?)\.)?([\w\.\-]+)(?:[\\\/](?:.+))?

MONEY: \$([0-9]{1,3}(?:(?:\,[0-9]{3})+)?(?:\.[0-9]{1,2})?)\s

Turning text into data
Pre-processing

- Great, but we can't write patterns for everything
- Words are messy and have a lot of variation
- We need to collapse semantically
- We need to *clean / pre-process*

	Raw Documents	
1	Text analysis is fun	
2	I enjoy analyzing text data	
3	Data science often involves text analytics	

Turning text into data
Pre-processing

- Common first steps:
  - Spell check / correct
  - Remove punctuation / expand contractions

can’t -> cannot
they’re -> they_are
doesn’t -> does_not

	Raw Documents	**Processed Documents**
1	Text analysis is fun	
2	I enjoy analyzing text data	
3	Data science often involves text analytics	

Turning text into data
Pre-processing

- Now to collapse words with the same meaning
- We do this with *stemming* or *lemmatization*
- Break words down to their roots

	Raw Documents	**Processed Documents**
1	Text analysis is fun	
2	I enjoy analyzing text data	
3	Data science often involves text analytics	

Turning text into data
Pre-processing

- Stemming is more conservative
- There are many different stemmers
- Here's the Porter stemmer (1979)

	Raw Documents	**Processed Documents**
1	Text analysis is fun	Text analysi is fun
2	I enjoy analyzing text data	I enjoy analyz text data
3	Data science often involves text analytics	Data scienc often involv text analyt

Turning text into data
Pre-processing

- Stemming is more conservative
- There are many different stemmers
- Here's the Porter stemmer (1979)

	Raw Documents	**Processed Documents**
1	Text analysis is fun	Text **analysi** is fun
2	I enjoy analyzing text data	I enjoy **analyz** text data
3	Data science often involves text analytics	Data scienc often involv text **analyt**

Turning text into data
Pre-processing

- The Lancaster stemmer (1990) is newer and more aggressive
- Truncates words a LOT

	Raw Documents	**Processed Documents**
1	Text analysis is fun	text **analys** is fun
2	I enjoy analyzing text data	I enjoy **analys** text dat
3	Data science often involves text analytics	dat sci oft involv text **analys**

Turning text into data
Pre-processing

- Lemmatization uses linguistic relationships and parts of speech to collapse words down to their root form - so you get actual words (“lemma”), not stems
- WordNet Lemmatizer

	Raw Documents	**Processed Documents**
1	Text analysis is fun	text analysis is fun
2	I enjoy analyzing text data	I enjoy analyze text data
3	Data science often involves text analytics	data science often involve text analytics

Turning text into data
Pre-processing

- Picking the right method depends on how much you want to preserve nuance or collapse meaning
- We'll stick with Lancaster

	Raw Documents	**Processed Documents**
1	Text analysis is fun	text analys is fun
2	I enjoy analyzing text data	I enjoy analys text dat
3	Data science often involves text analytics	dat sci oft involv text analys

Turning text into data
Pre-processing

- Finally, we need to remove words that don't hold meaning themselves
- These are called "stopwords"
- Can expand standard stopword lists with custom words

	Raw Documents	**Processed Documents**
1	Text analysis is fun	text analys fun
2	I enjoy analyzing text data	enjoy analys text dat
3	Data science often involves text analytics	dat sci oft involv text analys

Turning text into data
Pre-processing

- **A word of caution: there aren't any universal rules for making pre-processing decisions**
- Do what makes sense for your data - but be cautious of the researcher degrees of freedom involved
- See:
  - Denny and Spirling, 2016. Assessing the Consequences of Text Pre-processing Decisions"
  - Denny and Spirling, 2018. "Text Preprocessing for Unsupervised Learning: Why It Matters, When It Misleads, and What to Do About It"

Turning text into data
Tokenization

- Now we need to tokenize
- Break words apart according to certain rules
- Usually breaks on whitespace and punctuation
- What's left are called "tokens"
- Single tokens or pairs of two or more tokens are called "ngrams"

Turning text into data
Tokenization

- We can express the presence of each “ngram” as a column
- This is often called a “term frequency matrix”
- Here are unigrams

text	analys	fun	enjoy	dat	sci	oft	involv
1	1	1					
1	1		1	1			
1	1			1	1	1	1

Turning text into data
Tokenization

- We can express the presence of each “ngram” as a column
- This is often called a “term frequency matrix”
- And here are bigrams

text analys	analys fun	enjoy analys	analys text	text dat	dat sci	sci oft	oft involv
1	1						
		1	1	1			
1					1	1	1

Turning text into data
Tokenization

- If we want to characterize the whole corpus, we can just look at the most frequent words
- Here's the "term frequency matrix":

text	analys	fun	enjoy	dat	sci	oft	involv
1	1	1					
1	1		1	1			
1	1			1	1	1	1
3	**3**	**1**	**1**	**2**	**1**	**1**	**1**

Turning text into data
TF-IDF

- We already know these documents are about text analysis
- What if we want to distinguish documents from each other?
- What makes them *unique*?

Image credit: https://medium.com/@imamun/creating-a-tf-idf-in-python-e43f05e4d424

Turning text into data
TF-IDF

- Divide word frequencies by the number of documents they appear in
- Down-weight words that are common; log-scale emphasizes unique words
- Several variants that add smoothing

$$\text{tf-idf} = \text{tf} \times \text{idf} \qquad (1)$$

$$\text{idf}(t) = \log \frac{n+1}{\text{df}(d,t)+1} + 1 \qquad (2)$$

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

tf_{ij} = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Image credit: https://sites.temple.edu/tudsc/2017/03/30/measuring-similarity-between-texts-in-python/tfidf-equations/
Image credit: https://medium.com/@imamun/creating-a-tf-idf-in-python-e43f05e4d424

Turning text into data
TF-IDF

- The overall distribution of words is still largely preserved
- But now we're emphasizing what makes each document unique

text	analys	fun	enjoy	dat	sci	oft	involv
1	1	2.1					
1	1		2.1	1.4			
1	1			1.4	2.1	2.1	2.1
3	**3**	**2.1**	**2.1**	**2.8**	**2.1**	**2.1**	**2.1**

Turning text into data
TF-IDF

- The overall distribution of words is still largely preserved
- But now we're emphasizing what makes each document unique
- Within each document, we're now highlighting distinctive terms

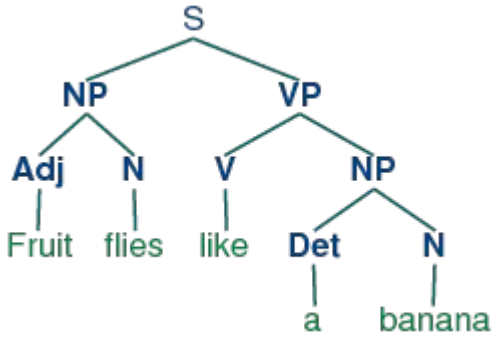
text	analys	fun	enjoy	dat	sci	oft	involv
1	1	2.1					
1	1		2.1	1.4			
1	1			1.4	2.1	2.1	2.1
3	**3**	**2.1**	**2.1**	**2.8**	**2.1**	**2.1**	**2.1**

Turning text into data
TF-IDF

- TF-IDF is an extremely common and useful way to convert text into useful quantitative features
- It’s often all you need
- But there are other, more complex ways to quantify text

Turning words into numbers
Part-of-Speech Tagging

- Sometimes you care about how a word is used
- Can use pre-trained **part-of-speech (POS) taggers**
- Can also help with things like negation
  - “Happy” vs. “NOT happy”




```
>>> text = word_tokenize("And now for something completely different")
>>> nltk.pos_tag(text)
[('And', 'CC'), ('now', 'RB'), ('for', 'IN'), ('something', 'NN'),
('completely', 'RB'), ('different', 'JJ')]
```

Image credit: http://nltk.sourceforge.net/doc/en/ch03.html
Image credit: https://www.nltk.org/book_1ed/ch05.html

Turning words into numbers
Named Entity Extraction

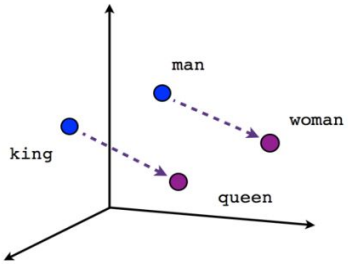
- Might also be interested in people, places, organizations, etc.
- Like POS taggers, **named entity extractors** use trained models

Figure 1: An example of NER application on an example text

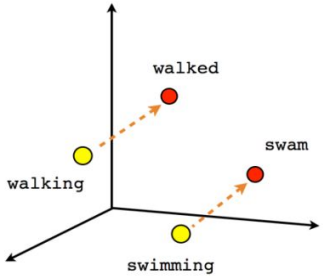
Image credit: http://inspiratron.org/blog/2019/04/15/building-named-entity-recognizer-ner-using-conditional-random-fields-crf/

Turning words into numbers
Word Embeddings

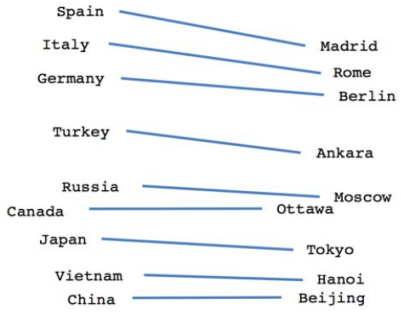
- Other methods can quantify words not by frequency, but by their relation
- **Word2vec** uses a sliding window to read words and learn their relationships; each word gets a vector in N-dimensional space
- Pretrained model: https://code.google.com/archive/p/word2vec/



Male-Female



Verb tense



Country-Capital

Image credit: https://www.tensorflow.org/tutorials/text/word_embeddings

Analysis
Finding patterns in text data

Finding patterns in text data

Two types of approaches:

- Unsupervised NLP: automated, extracts structure from the data
  - Clustering
  - Topic modeling
  - Mutual information
- Supervised NLP: requires training data, learns to predict labels and classes
  - Classification
  - Regression

Finding patterns in text data
Unsupervised methods

Collocation / phrase detection

- Simple way to get a quick look at common themes
- Bigrams are a form of “collocation” – a more general term for words that occur together

```
In [12]: import nltk, re
         from nltk.collocations import *
         bigram_measures = nltk.collocations.BigramAssocMeasures()
         tokens = nltk.corpus.genesis.words('english-web.txt')
         tokens = [re.sub(r'\W', '', t) for t in tokens]
         finder = BigramCollocationFinder.from_words(tokens)
         ignored_words = nltk.corpus.stopwords.words('english')
         finder.apply_word_filter(lambda w: len(w) < 3 or w.lower() in
         ignored_words)
         scored = finder.score_ngrams(bigram_measures.raw_freq)
         sorted(bigram for bigram, score in scored)[:10]

Out[12]: [(u'Abel', u'Mizraim'),
          (u'Abel', u'also'),
          (u'Abimelech', u'called'),
          (u'Abimelech', u'charged'),
          (u'Abimelech', u'king'),
          (u'Abimelech', u'rose'),
          (u'Abimelech', u'said'),
          (u'Abimelech', u'took'),
          (u'Abimelech', u'went'),
          (u'Abraham', u'answered')]
```

Code modified from: https://www.nltk.org/howto/collocations.html

Finding patterns in text data
Unsupervised methods

Co-occurrence matrices

- We can also find words that occur in the same documents together (not just next to each other)

```
from sklearn.feature_extraction.text import CountVectorizer
count_vectorizer = CountVectorizer(
    max_df=.9, min_df=50, binary=True, stop_words=stop_words
)
counts = count_vectorizer.fit_transform(sample['Text'])
ngrams = count_vectorizer.get_feature_names()
cooccurs = (counts.T * counts)
cooccurs.setdiag(0)
cooccurs = pd.DataFrame(cooccurs.todense(), index=ngrams, columns=ngrams)

cooccurs = cooccurs.where(np.triu(np.ones(cooccurs.shape)).astype(np.bool))
cooccurs = cooccurs.stack().reset_index()
cooccurs.columns = ['ngram1', 'ngram2', 'count']
cooccurs.sort_values("count", ascending=False).head(5)
```

	able	absolutely	acid	actual	actually	add
able	0	10	4	3	30	19
absolutely	10	0	2	6	23	17
acid	4	2	0	1	10	12
actual	3	6	1	0	14	8
actually	30	23	10	14	0	33

5 rows × 1026 columns

	ngram1	ngram2	count
315802	good	taste	1000.0
321607	great	taste	796.0
290399	flavor	taste	788.0
289873	flavor	good	756.0
315600	good	product	656.0

Finding patterns in text data
Unsupervised methods

- Might want to compare documents (or words) to one another
- Possible applications
  - Spelling correction
  - Document deduplication
  - Measure similarity of language
    - Politicians' speeches
    - Movie reviews
    - Product descriptions

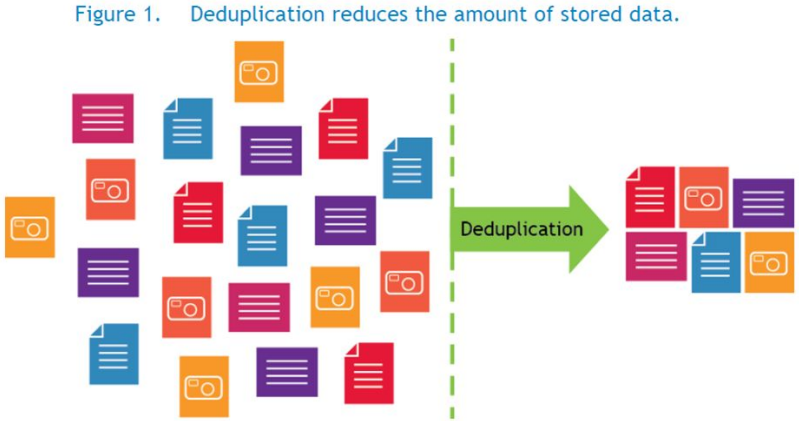


Image credit: https://pibytes.wordpress.com/2013/02/02/deduplication-internals-part-1/

Finding patterns in text data
Unsupervised methods

Levenshtein distance

- Compute number of steps needed to turn a word/document into another
- Can express as a ratio (percent of word/document that needs to change) to measure similarity

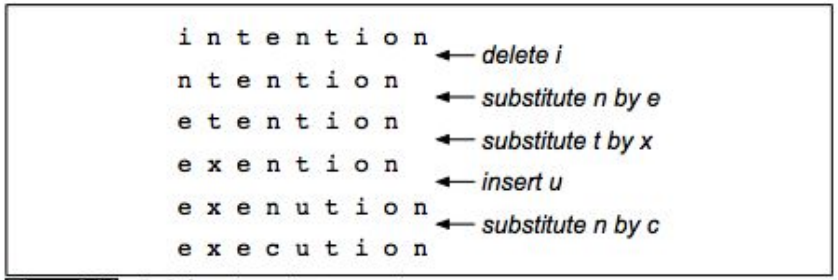


Figure 2.16 Path from *intention* to *execution*.

Image credit: http://web.stanford.edu/~jurafsky/slp3/2.pdf

Finding patterns in text data
Unsupervised methods

Cosine similarity

- Compute the “angle” between two word vectors
- TF-IDF: axes are the weighted frequencies for each word
- Word2Vec: axes are the learned dimensions from the model

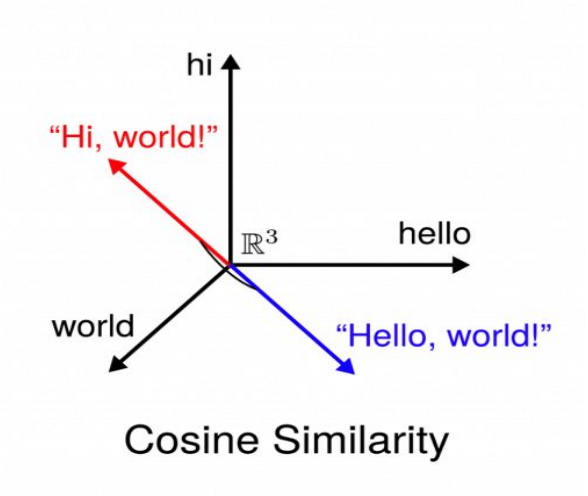


Image credit: https://www.machinelearningplus.com/nlp/cosine-similarity/

Finding patterns in text data
Unsupervised methods

Clustering

- Algorithms that use word vectors (TF-IDF, Word2Vec, etc.) to identify structural groupings between observations (words, documents)
- K-Means is a very commonly used one

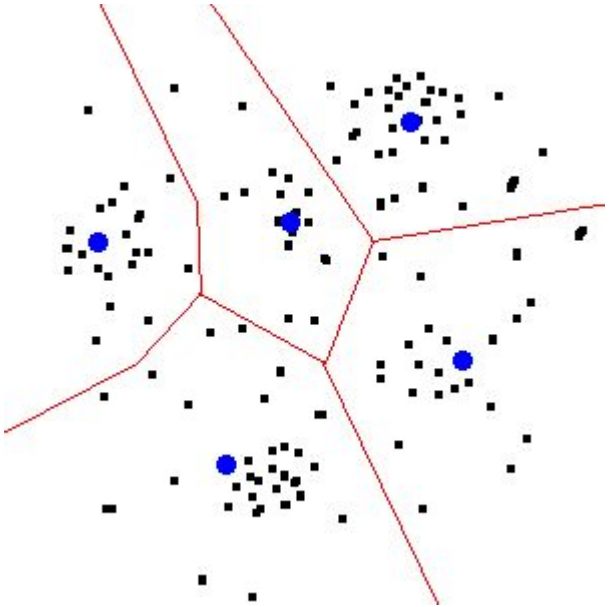
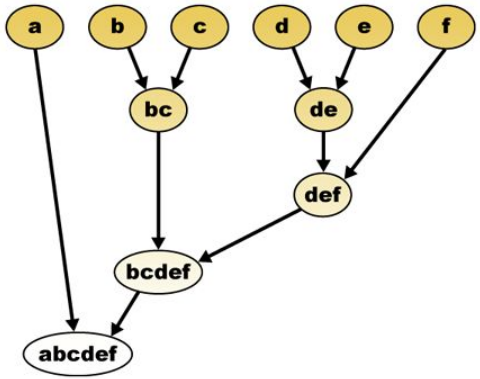


Image credit: http://mnemstudio.org/clustering-k-means-introduction.htm

Finding patterns in text data
Unsupervised methods

Hierarchical/agglomerative clustering

- Start with all observations, and use a rule to pair them up, and repeat until there's only one group left



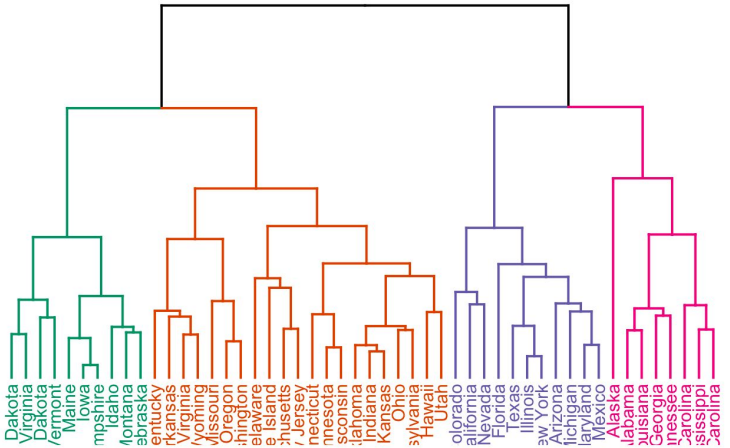


Image credit: https://scrnaseq-course.cog.sanger.ac.uk/website/biological-analysis.html
Image credit: https://rpkgs.datanovia.com/factoextra/reference/fviz_dend.html

Finding patterns in text data
Unsupervised methods

Network analysis

- Can also get creative
- After all, we're just working with columns and numbers
- Example: link words together by their strongest correlations

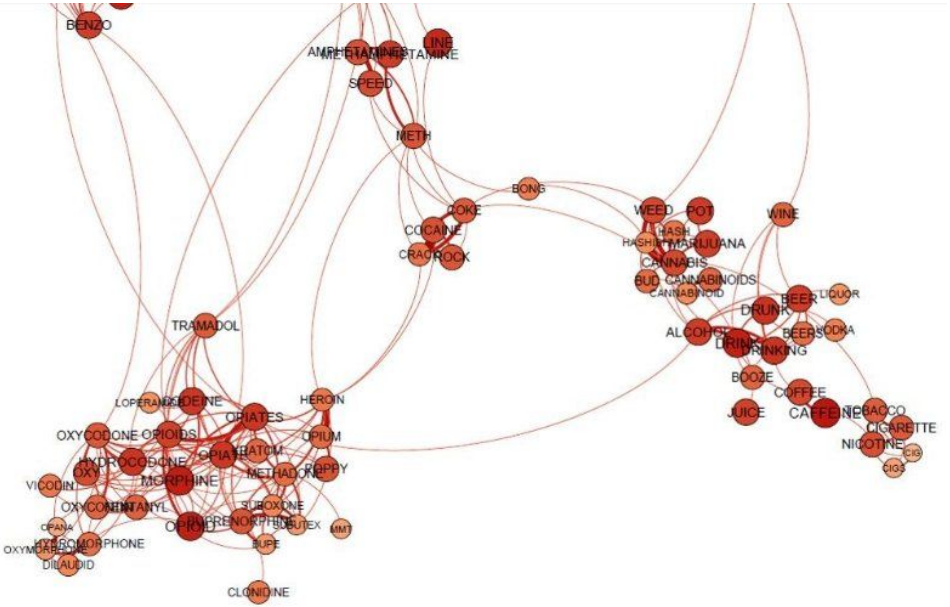


Image credit: https://www.linkedin.com/in/patrick-van-kessel

Finding patterns in text data
Unsupervised methods

Pointwise mutual information

- Based on information theory
- Compares conditional and joint probabilities to measure the likelihood of a word occurring with a category/outcome, beyond random chance

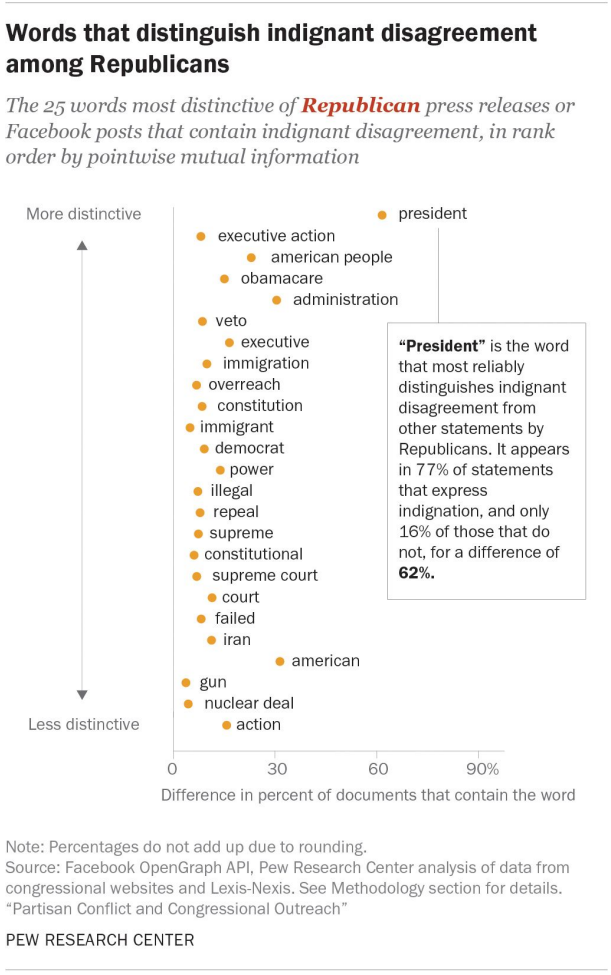


Image credit: https://www.people-press.org/2017/02/23/partisan-language-in-congressional-outreach/

Finding patterns in text data
Unsupervised methods

Topic modeling

- Algorithms that characterize documents in terms of topics (groups of words)
- Find topics that best fit the data

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Image credit: http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

Finding patterns in text data
Supervised methods

- Often we want to categorize documents
- Unsupervised methods can help
- But often we need to read and label them ourselves
- Classification models can take labeled data and learn to **make predictions**

Finding patterns in text data
Supervised methods

Steps:

- Label a sample of documents
- Break your sample into two sets: a **training sample** and a **test sample**
- **Train** a model on the training sample
- **Evaluate** it on the test sample
- **Apply** it to the full set of documents to make **predictions**

Finding patterns in text data
Supervised methods

- First you need to develop a codebook
- Codebook: set of rules for labeling and categorizing documents
- The best codebooks have clear rules for hard cases, and lots of examples
- Categories should be MECE: mutually exclusive and collectively exhaustive

PLEASE EXPLORE THIS INTERFACE AND READ THROUGH ALL OF THE INSTRUCTIONS BEFORE SUBMITTING YOUR FIRST HIT.
Hover over each prompt to view detailed explanations of each question, and click to view specific examples and pointers. It is very important that you follow the instructions carefully.

Author: Bernie Sanders
Party: Democratic Party
State: Vermont
Date: March 30, 2016 (Barack Obama was President)
Note: the above info is only for context, the actual post is below. Please use all of the content below to make your decisions (except for words contained in URLs/links)

Facebook post by Bernie Sanders on March 30, 2016

Message: Bernie Sanders:
Senator from Vermont?
Does not take money from super PACs?
Not for sale?
Wants to overturn Citizens United?
Thinks education and health care should be a right?
Democratic Socialist?
Doesn't want people to eat cat food?

Thanks Sarah!

Title: Sarah Silverman

Story: Bernie Sanders shared Sarah Silverman's video.

Description: Friendos! I made this vid about why I'm voting #BERNIE. Hope u eat it up.

Notes

Needs Review / Uncodeable

Does the post mention any of the following groups or institutions (NOT THE AUTHOR), and if so, does it express any support and/or opposition?

	Mentioned?	Supports / Agrees	Opposes / Disagrees	Angry or Insulting?
Donald Trump, his administration, or his campaign	☐			
Barack Obama or his administration	☐			
Hillary Clinton or her campaign	☐			
Federal agencies	☐			
Republicans, 'conservatives', or conservative values	☐			
Democrats, 'liberals', or liberal values	☐			

ENGAGE: Does the post encourage the reader to like, share, comment on, read, listen to, or watch something? ☐

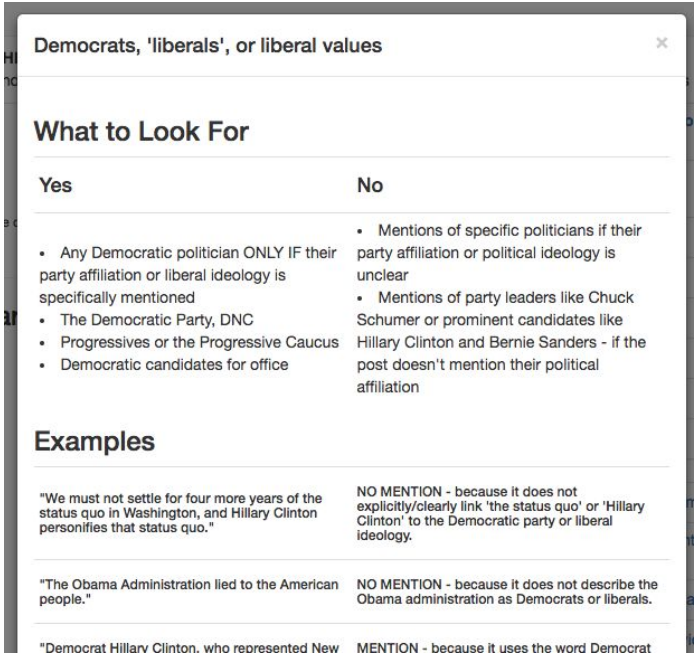
POLITICAL ACTION: Does the post invite the reader to vote, volunteer, call or send messages, sign a petition, attend an event/rally/protest, or make a donation? ☐

ELECTION-RELATED: Does the post mention specific elections, campaigns, or candidates? ☐

LOCAL REFERENCE: Does the post mention a place, group, individual(s), or event in the politician's state or district? ☐

Submit

Finding patterns in text data
Supervised methods



Finding patterns in text data
Supervised methods

- Need to validate the codebook by measuring **interrater reliability**
- Makes sure your measures are consistent, objective, and reproducible
- Multiple people code the same document

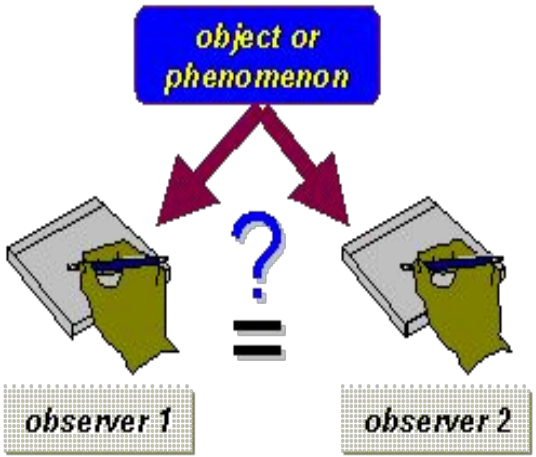


Image credit: https://socialresearchmethods.net/kb/reltypes.php

Finding patterns in text data
Supervised methods

- Various metrics to test whether their agreement is high enough
  - Krippendorf's alpha
  - Cohen's kappa
- Can also compare coders against a gold standard, if available

Values	Interpretation
Smaller than 0.00	Poor Agreement
0.00 to 0.20	Slight Agreement
0.21 to 0.40	Fair Agreement
0.41 to 0.60	Moderate Agreement
0.61 to 0.80	Substantial Agreement
0.81 to 1.00	Almost Perfect Agreement

Image credit: https://www.researchgate.net/figure/Interpretation-of-Cohens-Kappa-Values_tbl2_302869046

Finding patterns in text data
Supervised methods

- Mechanical Turk can be a great way to code a lot of documents
- Have 5+ Turkers code a large sample of documents
- Collapse them together with a rule
- Code a subset in-house, and compute reliability

Finding patterns in text data
Supervised methods

- Sometimes you're trying to measure something that's really rare
- Consider **oversampling**
- Example: keyword oversampling
- Disproportionately select documents that are more likely to contain your outcomes

Finding patterns in text data
Supervised methods

- After coding, split your sample into two sets (~80/20)
  - One for training, one for testing
- We do this to check for (and avoid) **overfitting**

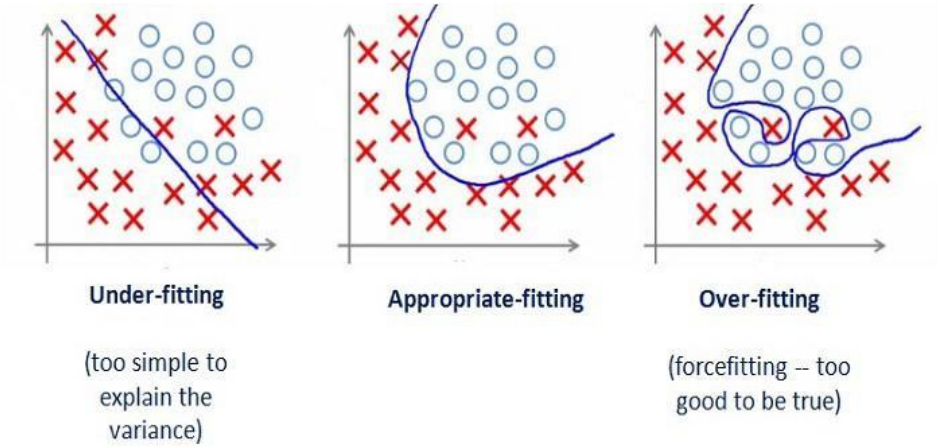


Image credit: https://medium.com/ml-research-lab/under-fitting-over-fitting-and-its-solution-dc6191e34250

Finding patterns in text data
Supervised methods

- Next step is called **feature extraction** or **feature selection**
- Need to extract “features” from the text
  - TF-IDF
  - Word2Vec vectors
- Can also utilize metadata, if potentially useful

Finding patterns in text data
Supervised methods

- Select a classification algorithm
- Common choice for text data are **support vector machines (SVMs)**
- Similar to regression, SVMs find the line that best separates two or more groups
- Can also use non-linear “kernels” for better fits (radial basis function, etc.)
- **XGBoost** is a newer and very effective algorithm

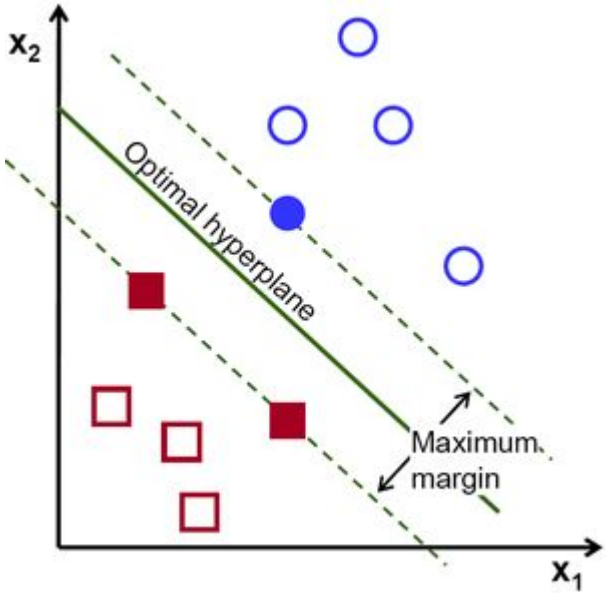


Image credit: https://towardsdatascience.com/support-vector-machine-vs-logistic-regression-94cc2975433f

Finding patterns in text data
Supervised methods

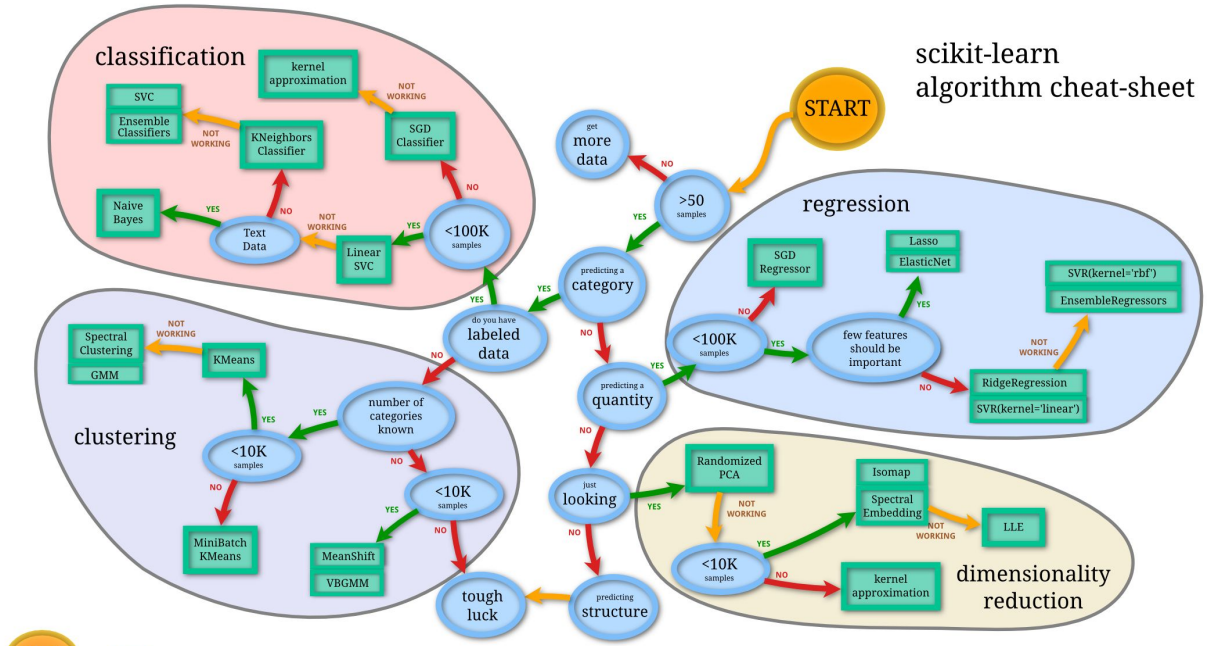


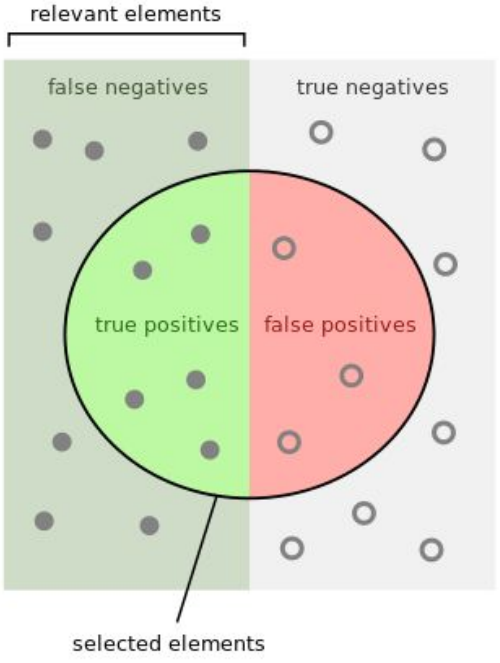
Image credit: https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

Finding patterns in text data
Supervised methods

- Word of caution: if you oversampled, your weights are probability (survey) weights
- Most machine learning implementations assume your weights are frequency weights
- Not an issue for predictions - but can produce biased variance estimates if you analyze your training data itself

Finding patterns in text data
Supervised methods

- Time to evaluate performance
- Lots of different metrics, depending on what you care about
- Often we care about precision/recall
  - Precision: did you pick out mostly needles or mostly hay?
  - Recall: how many needles did you miss?
- Other metrics:
  - Matthew's correlation coefficient
  - Brier score
  - Overall accuracy



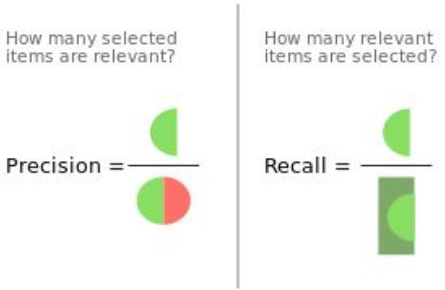


Image credit: https://en.wikipedia.org/wiki/Precision_and_recall

Finding patterns in text data
Supervised methods

- Doing just one split leaves a lot up to chance
- To bootstrap a better estimate of the model's performance, it's best to use **K-fold cross-validation**
- Splits your data into train/test sets **multiple times** and averages the performance metrics
- Ensures that you didn't just get lucky (or unlucky)

Finding patterns in text data
Supervised methods

- Model not working well?
- You probably need to **tune your parameters**
- You can use a **grid search** to test out different combinations of model parameters and feature extraction methods
- Many software packages can automatically help you pick the best combination to maximize your model's performance

Finding patterns in text data
Supervised methods

- Suggested design:
  - Large training sample, coded by Turkers
  - Small evaluation sample, coded by Turkers and in-house experts
  - Compute IRR between Turk and experts
  - Train model on training sample, use 5-fold cross-validation
  - Apply model to evaluation sample, compare results against in-house coders and Turkers

Finding patterns in text data
Supervised methods

- Some (but not all) models produce probabilities along with their classifications
- Ideally you fit the model using your preferred scoring metric/function
- But you can also use post-hoc probability thresholds to adjust your model's predictions

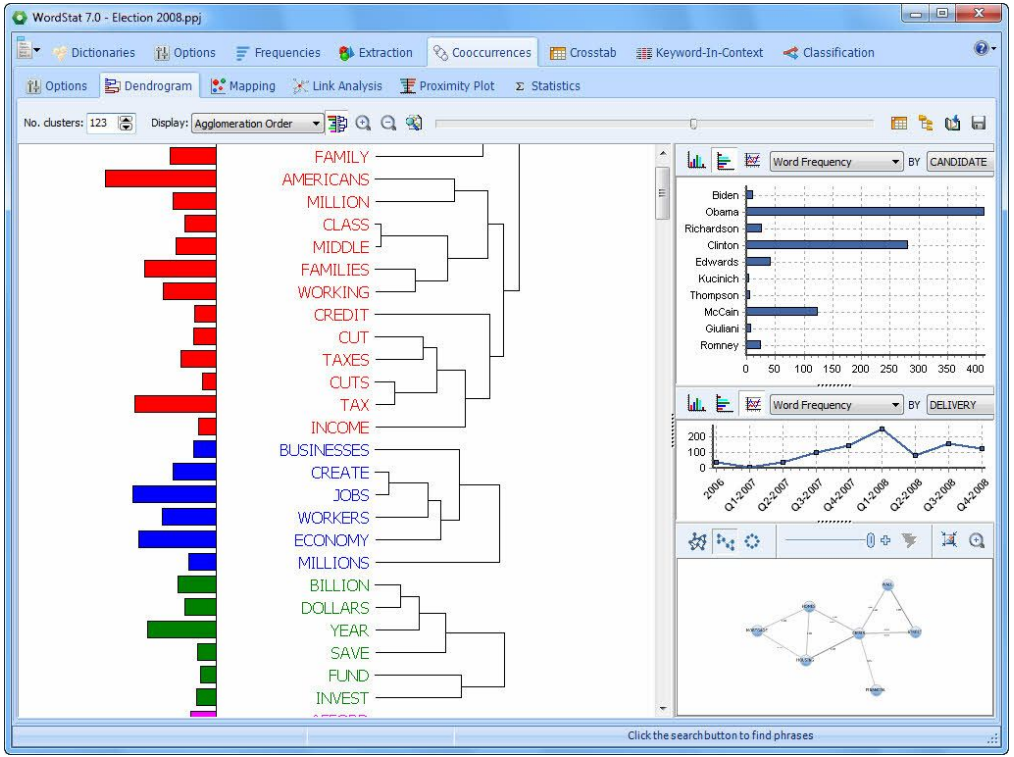
Tools and Resources

Open-source tools

- Python
  - NLTK, scikit-learn, pandas, numpy, scipy, gensim, spacy, etc
- R
  - tidytext, quanteda, tm, stm, tidymodels, textrecipes
  - https://cran.r-project.org/web/views/NaturalLanguageProcessing.html
- Java
  - Stanford Core NLP + many other useful libraries https://nlp.stanford.edu/software/

Commercial tools

- Cloud-based NLP
  - Amazon Comprehend
  - Google Cloud Natural Language
  - IBM Watson NLU
- Software
  - SPSS Text Modeler
  - Provalis WordStat



Thank you!

Notebook:
https://colab.research.google.com/github/patrickvankessel/text-analysis-workshop/blob/main/Tutorial.ipynb
GitHub repo: https://github.com/patrickvankessel/text-analysis-workshop

Feel free to reach out:
pvankessel@pewresearch.org
patrickvankessel@gmail.com

Special thanks to Mika Jugovich for help putting these materials together for previous workshops