Computational Data Analytics
for Economists

Lecture 3

Analysis of Text Data

Helge Liebert Anthony Strittmatter

Outline

Literature

Matthew Gentzkow, Bryan T. Kelly and Matt Taddy (2018)
Text as Data
Journal of Economic Literature, forthcoming

Dan Jurafsky and James H. Martin (2018)
Speech and Language Processing (3rd ed. draft)
`https://web.stanford.edu/~jurafsky/slp3/`

Introduction

- 90% of data on the Internet has been created in the last two years.
- 235 million emails sent per day.
- 3.3 million Facebook posts created every minute.
- 3.8 million Google searches performed each minute.
- 1.7 megabytes of new information created every second, per person.

➥ An immense amount of data (new and old) is recorded as text.

Introduction

- Text differs from other, traditional forms of data.
- Text is inherently *unstructured* and *high-dimensional*.
- One of the major fields of application of machine learning methods.
- Fast-growing field. Many new techniques developed in industry.
- Recent applications in economics and other social sciences.

Working with text data

- Whole fields of research in different disciplines are devoted to this.
- Historically distinct fields like natural language processing, speech recognition, computational linguistics are merging in recent years.
- Extremely fast-paced development due to many practitioners.
- Research is commercially viable, well-performing implementations may not be public domain.
- Broad label: ‘Speech and language processing’ (Jurafsky & Martin 1999, 2018).

Text as data

- Text is inherently high-dimensional.
- Example: A sample of 30-word Twitter documents using only the 1,000 most common words in the english language ($w = 30$, $p = 1,000$).
- Unique representation p^w: About as many dimensions as atoms in the universe.

➥ Requires thinking about how text can be represented as data.

Statistical models for text data

- Statistical methods used to analyze text are closely related to methods used to analyse high-dimensional data in other domains.
- Lasso/penalized regression applied more or less as in other settings.
- Other methods (e.g. topic models or multinomial inverse regression) have been adapted to the specific structure of text data.
- R vs. Python: Similar considerations as in last lecture.

Analysis summary

- In the cases considered, analysis can be summarized in three steps:
  1. Represent raw text $\mathcal{D}$ as a numerical array $\mathbf{C}$.
  2. Map $\mathbf{C}$ to predicted values of (unknown) outcomes $\mathbf{V}$.
  3. (Use $\hat{\mathbf{V}}$ in subsequent descriptive or causal analysis.)

Analysis summary

1. Represent raw text $\mathcal{D}$ as a numerical array $\mathbf{C}$.

- Pre-processing: Researcher must impose some preliminary restrictions to reduce the dimensionality of the data.
- No current technique can deal with 1000^{30}-dimensional Twitter data.
- In most cases, elements of $\mathbf{C}$ are counts of *tokens* (e.g. words, phrases, pre-defined features).
- Other approaches leverage prior information about the structure of language before any analysis to reduce dimensionality.
- Dimension likely to be large, possibly $p \gg n$.

Analysis summary

2. Map **C** to predicted values of (unknown) outcomes **V**.

- Involves application of (high-dimensional) statistical methods.
- Classical example: data is the text of emails, **V** is an indicator for whether the email is spam, prediction $\hat{\mathbf{V}}$ determines whether to flag email as spam.
- Alternatively: Sentiment prediction, predicting flu outbreaks from google searches, grouping texts by topic, stock prices, political slant.

Analysis summary

3. Use $\hat{\mathbf{V}}$ in subsequent descriptive or causal analysis.

- Goal is often prediction rather than causal inference; the interpretation of the mapping from $\mathbf{V}$ to $\hat{\mathbf{V}}$ is not usually of interest.
- Aim in social science is often to use $\hat{\mathbf{V}}$ to infer causal relationships or structural parameters.
- Optional and application-specific. Not covered here.

Representing text as data

- Humans read text in context, not vectors of binary variables or sequences of unrelated tokens.
- We interpret words in light of other words, extracting meaning from a text as a whole.
- Text analysis in social sciences and machine learning ignores most of this complexity.
- Still, computational linguistics has become pretty good at context interpretation tasks in recent years (mobile phone text entry, translations).

Representing text as data

- Typical simplifications when constructing **C**:
  (a) divide text into individual documents,
  (b) reduce the number of language elements considered,
  (c) limit the extent to which we encode dependence among elements within documents.
- Map raw text $\mathcal{D}$ to a numerical array **C**.
- Typically, a row c_i of **C** is a vector with each element indicating the presence or count of a particular language token j in document i.

Defining a document

- Divide raw text $\mathcal{D}$ into individual documents $\{\mathcal{D}_i\}$.
- Level often determined by the level at which **V** is defined.
- Choice not always clear. Finer partitions ease computation at the cost of limiting the dependence that can be captured.
- Most methods treat documents as independent.
- No theoretical guidance. Check sensitivity.
- For practical non-social science applications, individual documents may matter less.

Feature selection

- Restricting the set of tokens. Eases computation. Not all methods require or benefit from this.
- Strip elements from the raw text that are not words, removing punctuation, numbers, HTML tags, proper names, etc.
- Remove subsets of words that are very common or very rare.
- ‘Stop words’ are very common words, e.g. articles (‘the’, ‘a’), conjunctions (‘and’, ‘or’), forms of the verb ‘to be’, and more.
- Important for the grammatical structure but conveying little meaning on their own. Removal based on pre-defined lists is common practice.
- Very rare words do convey meaning, but the associated computational cost often exceeds their diagnostic value.
- Common practice is to exclude all terms that fewer than k times or in fewer than $k\%$ of documents for some arbitrary small k.

Feature selection: Filtering by tf-idf

- Combined approach: Term-frequency-inverse-document-frequency filtering.
- Term frequency tf_{ij}: The count c_{ij} of occurences of word or feature j in document i.
- Inverse document frequency idf_j: The log of one over the share of documents containing j, $\log(n/d_j)$ where $d_j = \sum_i \mathbb{1}\{c_{ij} > 0\}$.
- Filter words with low $tf_{ij} \times idf_j$ scores below some cutoff.
- Rare words have low scores because of low tf. Very common words that appear in most documents have low scores due to low idf.
- Keeps words that occur frequently in some documents but not in others.

Lemmatization

- Lemmatization is the task of determining that two words have the same root.
- Replace words with their root: '*economic*', '*economics*', '*economically*' are replaced by '*economic*'.
- Difficult, requires complicated morphological parsing algorithms.
- Morphological parser: '*cats*' parsed into two morphemes '*cat*' and '*s*'.
- Non-trivial: A Spanish word like '*amaren*' ('if in the future they would love') would need to be parsed into the morphemes *amar* ('to love'), *3PL*, and *future subjunctive*.

Lemmatization and stemming

- Stemming is a naive version of morphological analysis.
- Cut off word-final affixes based on a series of rewrite rules.

`ATIONAL`	$\longrightarrow$ `ATE`	(e.g., relational	$\longrightarrow$ relate)
`ING`	$\longrightarrow \epsilon$ *if stem contains vowel*	(e.g., motoring	$\longrightarrow$ motor)
`SSES`	$\longrightarrow$ `SS`	(e.g., grasses	$\longrightarrow$ grass)

Lemmatization and stemming

- Different stemming tools available, Porter (1980) has become standard for English and performs well in practice.
- Algorithm based on sequential rules. Errors of both over- and under-generalizing occur.
- Example input/output:

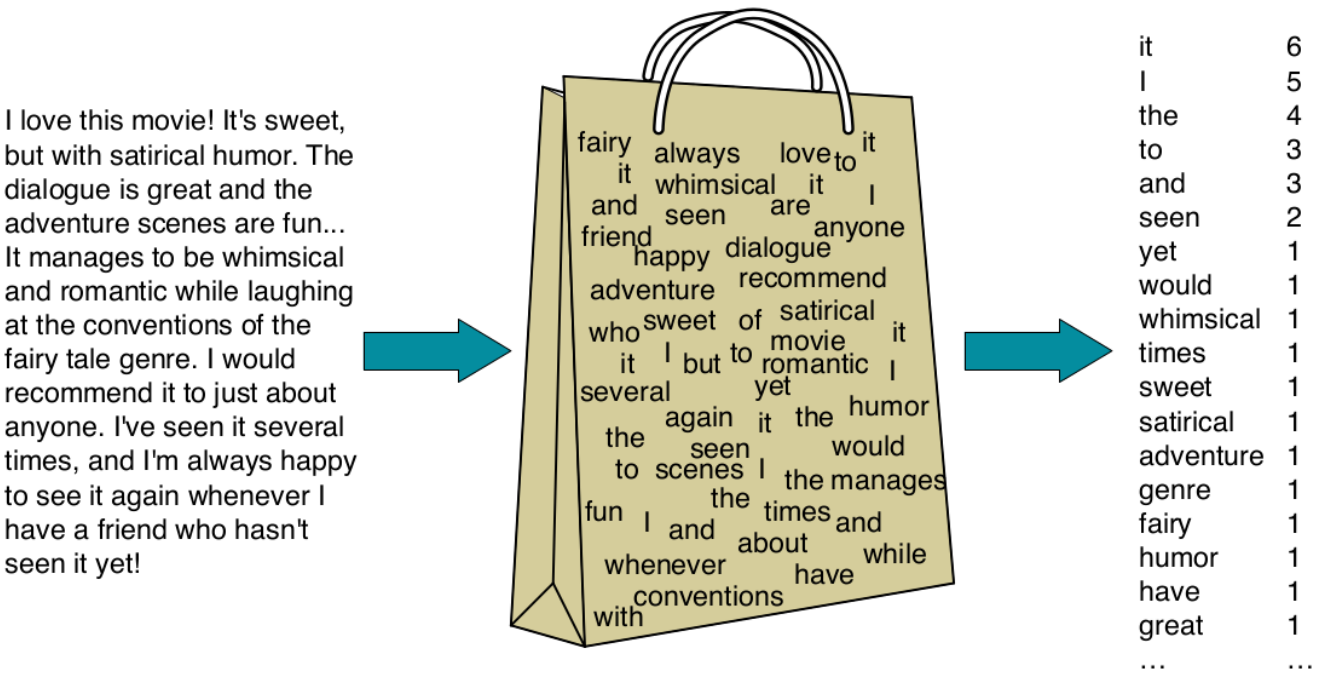
```
This was not the map we found in Billy Bones's chest, but
an accurate copy, complete in all things-names and heights
and soundings-with the single exception of the red crosses
and the written notes.

Thi wa not the map we found in Billi Bone s chest but an
accur copi complet in all thing name and height and sound
with the singl except of the red cross and the written note
```

Issues

- Cleaning helps reducing the number of unique language elements and the dimensionality of the data.
- Provides computational benefits and is often key to getting interpretable model fits (e.g. in topic modeling).
- But requires careful decisions about the elements likely to carry meaning in the particular application.
- ‘One researcher’s stop words are another’s subject of interest.’
- Context matters. Filtering may imply loss of information.
- Drop numerals from ‘the first 100 days’ or ‘September 11’? How about online communication, e.g. :-) ?

N-grams

- Limit dependence among language elements to get a tractable representation.
- Simplest approach: Bag-of-words/1-grams. Ignore word order.
- $\mathbf{c}_i$ is a vector of the length of the vocabulary contained in $\mathcal{D}$ and the elements c_{ij} are counts of occurence.



N-grams

Good night, good night! Parting is such sweet sorrow.

`good night good night part sweet sorrow`

1-grams
$\mathbf{c}_{ij} = 2$ for $j \in \{good, night\}$
$\mathbf{c}_{ij} = 1$ for $j \in \{part, sweet, sorrow\}$
$\mathbf{c}_{ij} = 0$ for all other words in the vocabulary

2-grams
$\mathbf{c}_{ij} = 2$ for $j \in \{good.night\}$
$\mathbf{c}_{ij} = 1$ for $j \in \{night.good, night.part, part.sweet, sweet.sorrow\}$
$\mathbf{c}_{ij} = 0$ for all other possible 2-grams

N-grams

- n-grams of order $n > 1$ yields data that captures a limited amount of dependence between words.
- n-gram counts sufficient for estimation of an n-order homogenous Markov model across words, i.e., a model that assumes word choice only depends upon the previous n words.
- Allows richer modeling where simple words may be insufficient to capture patterns of interest, e.g. partisan overtones in ‘death tax’ or ‘tax break’.
- But: dimension of $\mathbf{c}_i$ increases exponentially with n. More than 3-grams is rarely used. Return small relative to cost.
- Begin with bag-of-words, evaluate whether moving to larger n is worthwile.

Practical considerations

- Social science applications often diverge from ideal case considered in statistics.
- Level of aggregation differs.
- Full text sometimes not available, e.g. counts obtained through interface queries.
- Obtaining counts for large feature sets (all 2-Grams in the English language) may not be feasible.
- Initial feature selection necessarily aggressive—may not be inoccous.

Representation concepts: Term-document-matrix

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	5	117	0	0

Figure 15.1 The term-document matrix for four words in four Shakespeare plays. Each cell contains the number of times the (row) word occurs in the (column) document.

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	1	8	15
soldier	2	2	12	36
fool	37	58	1	5
clown	5	117	0	0

Figure 15.2 The term-document matrix for four words in four Shakespeare plays. The red boxes show that each document is represented as a column vector of length four.

Vector representation: Term-document-matrix

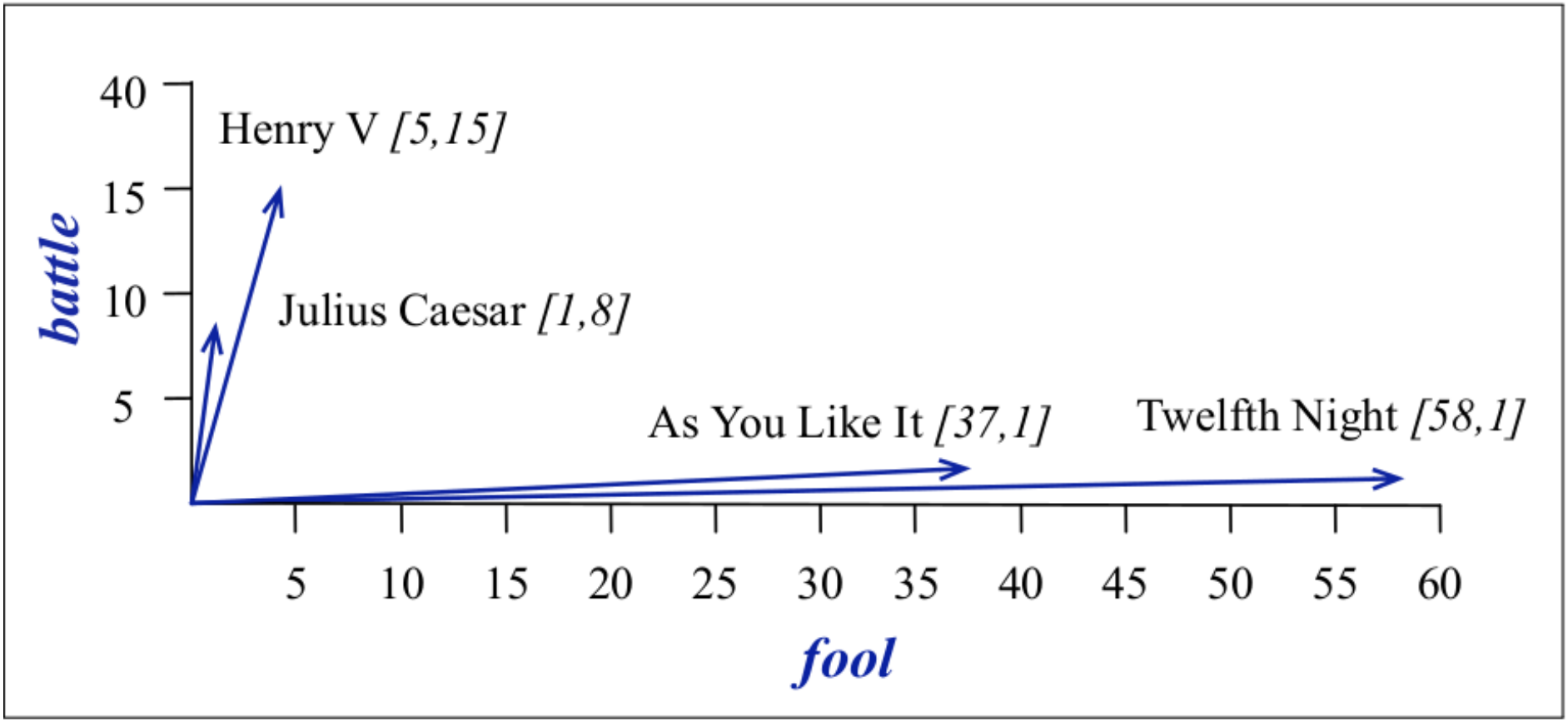


Figure 15.3 A spatial visualization of the document vectors for the four Shakespeare play documents, showing just two of the dimensions, corresponding to the words *battle* and *fool*. The comedies have high values for the *fool* dimension and low values for the *battle* dimension.

Term-term-matrix/term-context matrix

sugar, a sliced lemon, a tablespoonful of **apricot** jam, a pinch each of,
their enjoyment. Cautiously she sampled her first **pineapple** and another fruit whose taste she likened
well suited to programming on the digital **computer**. In finding the optimal R-stage policy from
for the purpose of gathering data and **information** necessary for the study authorized in the

	aardvark	...	computer	data	pinch	result	sugar	...
apricot	0	...	0	0	1	0	1	
pineapple	0	...	0	0	1	0	1	
digital	0	...	2	1	0	1	0	
information	0	...	1	6	0	4	0	

Figure 15.4 Co-occurrence vectors for four words, computed from the Brown corpus, showing only six of the dimensions (hand-picked for pedagogical purposes). The vector for the word *digital* is outlined in red. Note that a real vector would have vastly more dimensions and thus be much sparser.

Vector representation

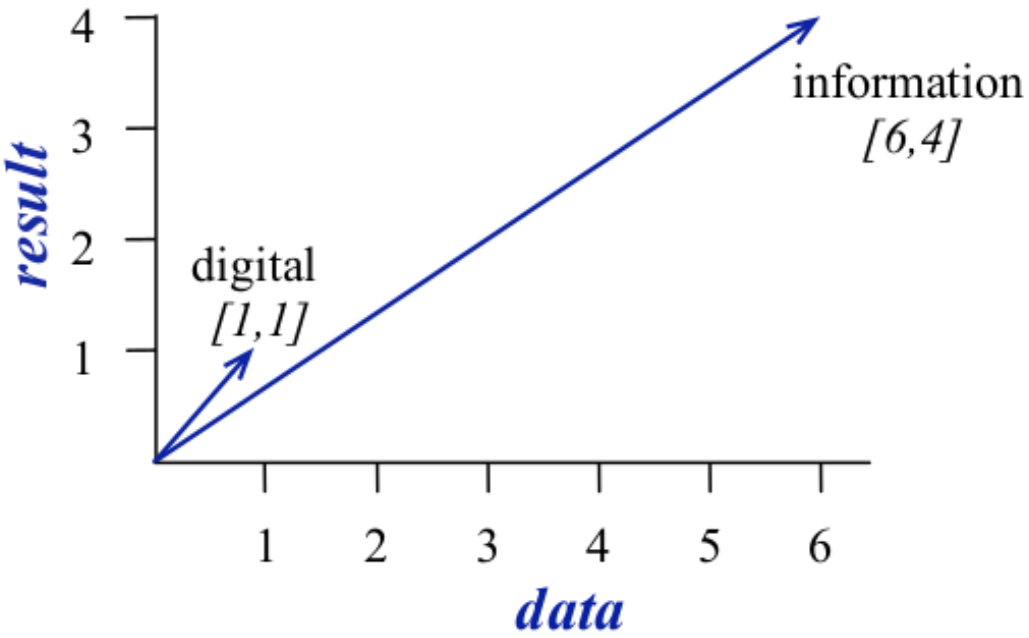


Figure 15.5 A spatial visualization of word vectors for *digital* and *information*, showing just two of the dimensions, corresponding to the words *data* and *result*.

Vector semantics

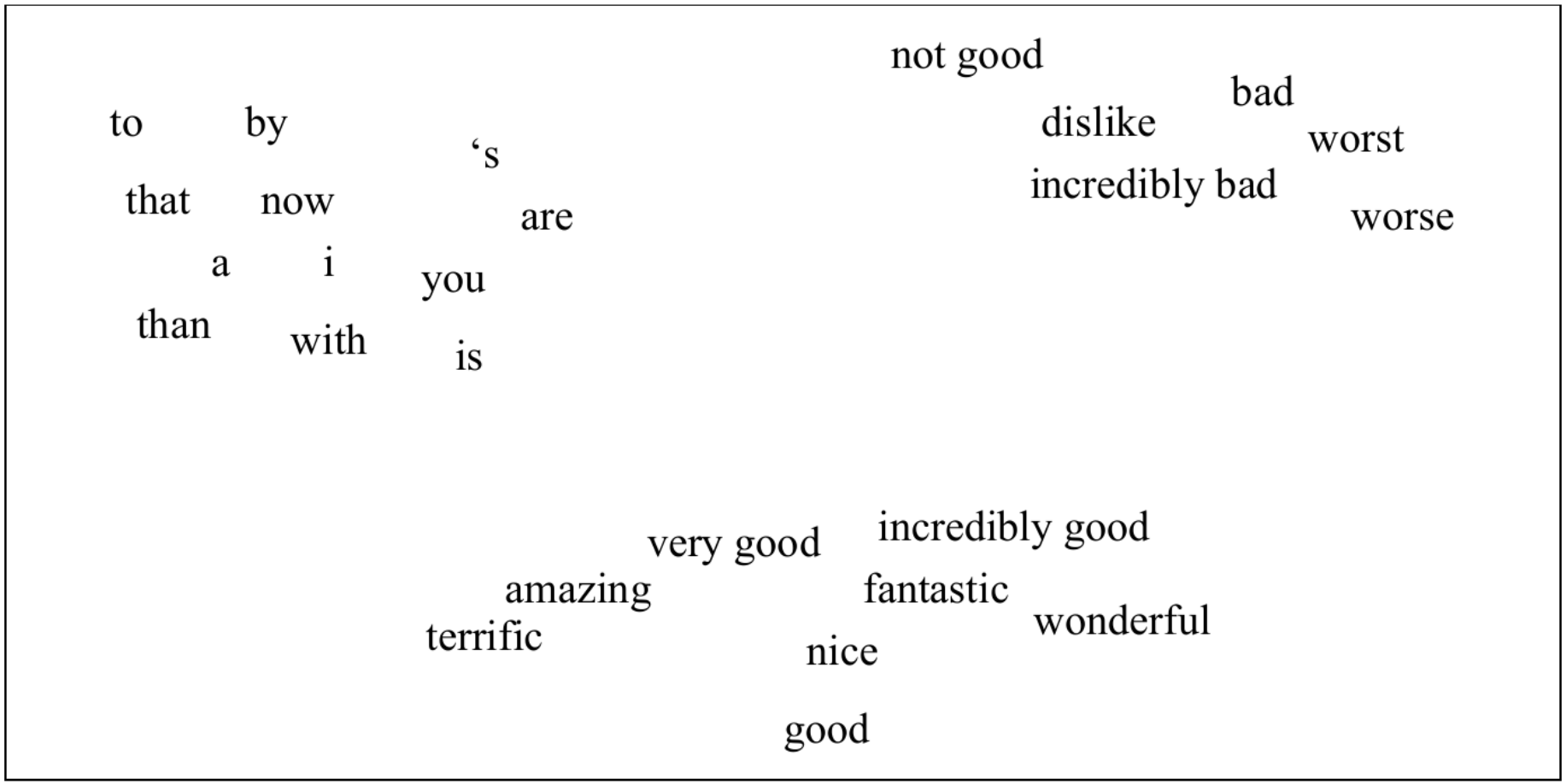


Figure 6.1 A two-dimensional (t-SNE) projection of embeddings for some words and phrases, showing that words with similar meanings are nearby in space. The original 60-dimensional embeddings were trained for a sentiment analysis task. Simplified from Li et al. (2015).

Vector semantics

- Disregard documents for now: How to best represent the meaning of words?
- Words can be represented by co-occurence vectors.
- Allows measuring similarity and relatedness between words (dot product).
- Each word represented with a vector that is **long** (length $|\mathcal{V}|$, commonly 20,000–50,000) and **sparse**.
- Difficult to work with.

Semantics with dense vectors

- Find an alternative word representation using vectors that are **short** (of length 50–300) and **dense**.
- Word embeddings—represent a word as being embedded in a vector space.
- Powerful representation: words that have similar meanings or grammatical properties have similar vectors.
- vector(king) - vector(man) + vector(woman) $\approx$ vector(queen)

Semantics with dense vectors

- Suppose this sentence occured in our training set:
  "`I have to make sure when I get home to feed the cat.`"
- We are trying to predict what comes after:
  "`I forgot when I got home to feed the cat.`"
- Suppose we've never seen the word "dog" after the words "feed the".
- Traditional N-gram model will predict cat and won't expect "dog".
- A word embedding model will assign a reasonably high probability to "dog" as well as "cat", merely because they have similar vectors.

Vector semantics: Relational properties

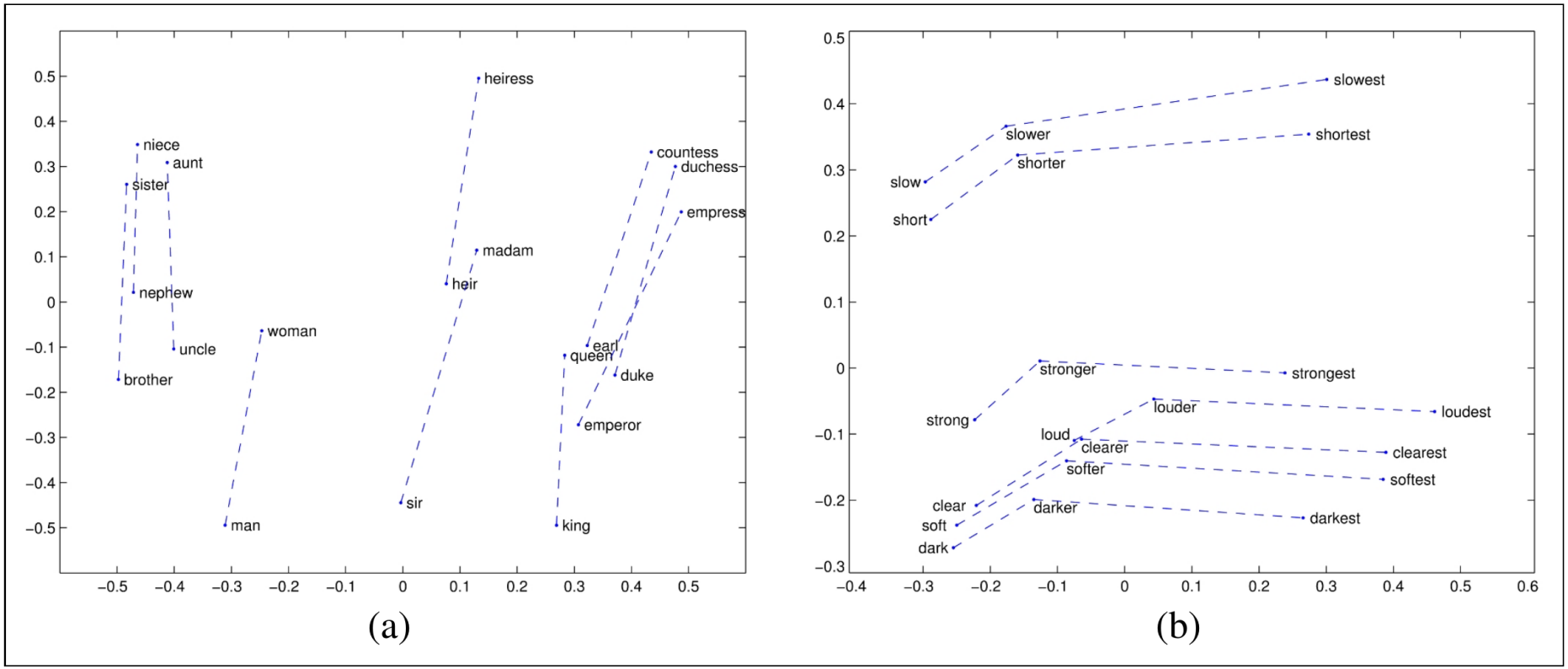


Figure 6.14 Relational properties of the vector space, shown by projecting vectors onto two dimensions. (a) 'king' - 'man' + 'woman' is close to 'queen' (b) offsets seem to capture comparative and superlative morphology (Pennington et al., 2014).

Vector semantics: Meaning over time

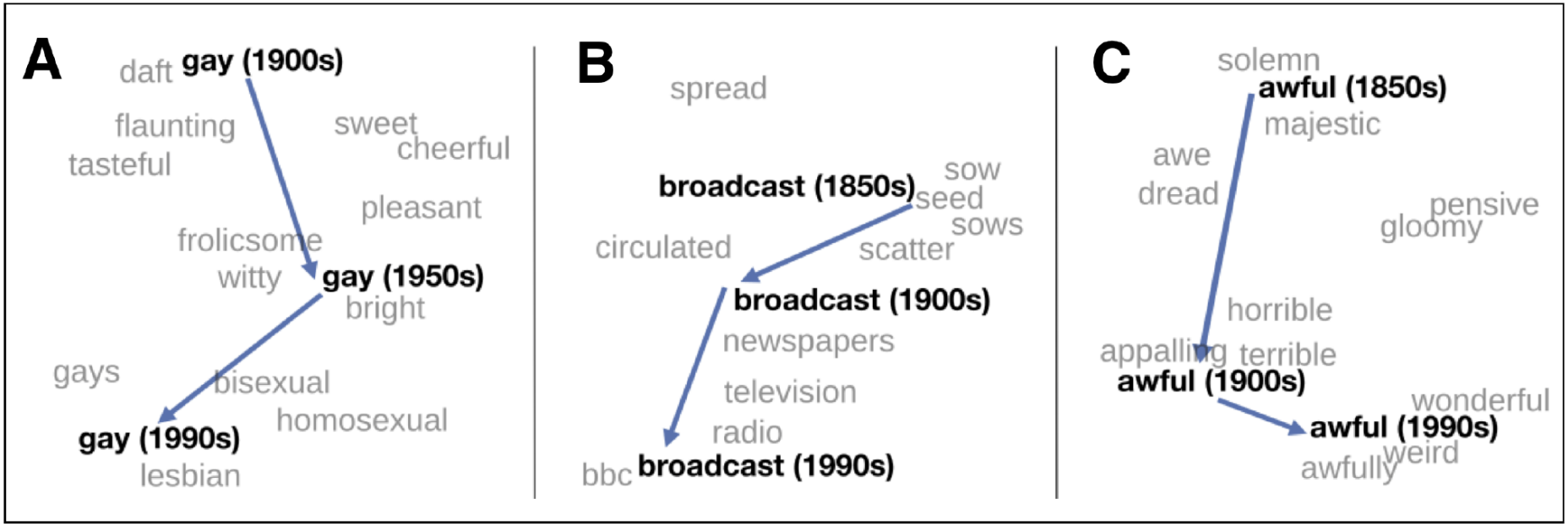


Figure 6.15 A t-SNE visualization of the semantic change of 3 words in English using word2vec vectors. The modern sense of each word, and the grey context words, are computed from the most recent (modern) time-point embedding space. Earlier points are computed from earlier historical embedding spaces. The visualizations show the changes in the word *gay* from meanings related to "cheerful" or "frolicsome" to referring to homosexuality, the development of the modern "transmission" sense of *broadcast* from its original sense of sowing seeds, and the pejoration of the word *awful* as it shifted from meaning "full of awe" to meaning "terrible or appalling" (Hamilton et al., 2016b).

Word embeddings

- Better to include as features in machine learning methods.
- Common dimension reduction methods: Singular value decomposition (e.g. PCA, factor analysis), neural net/word2vec (skip-gram, CBOW), GloVe.
- Unsupervised: Embeddings can be learned from any text.
- word2vec: Use running text as implicitly supervised training data. Sample from text to estimate a classifier predicting how often word y occurs near word x. Classifier weights used as word embeddings.
- GloVe: Relies on matrix factorization of the co-occurence statistics.
- Learning embeddings is pointless without substantial data.
- But plenty of pre-trained models are available (e.g. GloVe, fasttext).
- Unless your text is very special, these are fine to work with. Trained on Wikipedia, Common Crawl or Twitter.

Word embeddings

- Embeddings capture synonymy better, but interpretability is lost.
- Differences can still be interpreted. Allows investigating e.g. gender stereotypes or measure implicit associations.
- Bolukbasi et al. (2016) look at embeddings from news texts. Closest occupation to ‘computer programmer’ - ‘man’ + ‘woman’ is ‘homemaker’. ‘Father’ is to ‘doctor’ as ‘mother’ is to ‘nurse’.
- Many (social science) applications still require aggregation at the document level. Doc2vec? Different possiblities, active development.
- Document embeddings could be used as input for selection-on-observables methods. Requires substantial data.

Statistical methods

- Methods for mapping the document-token matrix $\mathbf{C}$ to predictions $\hat{\mathbf{V}}$ of an atttribute $\mathbf{V}$.
- Data may be partitioned such that $\mathbf{C}^{train}$ (dim. $n^{train} \times p$) collects rows for which $\mathbf{V}^{train}$ (dim. $n^{train} \times k$) of $\mathbf{V}$ is observed.
- k is the number of attributes to predict.
- Attributes in $\mathbf{V}$ can be observable quantities (flu cases, movie review, unemployment rate) or latent (topics being discussed in politics or news).

Statistical methods

Four broad categories of models connecting counts c_i to attributes v_i:

1. Dictionary-based methods (no statistical model).
2. Text regression (beginning from a model of $p(v_i|c_i)$).
3. Generative models (beginning from a model of $p(c_i|v_i)$, supervised or unsupervised).
4. Deep learning techniques (neural networks, distributed language models leveraging richer text representation than token counts).

Dictionary methods

- No statistical inference at all.
- Specify $\hat{\mathbf{v}}_i = f(\mathbf{c}_i)$ for some known function $f(\cdot)$.
- The most common method in the social science literature so far.
- Popular for sentiment analysis.
- $f(\cdot)$ defined based on pre-defined or ad-hoc categorization, e.g. using a dictionary of word lists associated with sentiment categories.
- 'Standard' dictionaries are easily accessible in R.
- Be methodical if you develop your own dictionary mapping.

Text regression

- Start with a model for $p(\mathbf{v}_i|\mathbf{c}_i)$. Training data is available.
- Predicting $\mathbf{v}_i$ from $\mathbf{c}_i$ is a standard regression problem.
- OLS is infeasible.
- High dimensionality of $\mathbf{c}_i$ ($p \geq n^{train}$) requires appropriate techniques.
- Mainly applications of standard high-dimensional regression methods to text.
- Penalized linear (index) models popular.

Text regression

- Penalized linear (index) model:

$$E(\mathbf{v}_i|\mathbf{x}_i) = f(\boldsymbol{\eta}_i)$$

  where $\boldsymbol{\eta}_i = \alpha + \mathbf{x}_i'\boldsymbol{\beta}$ is a linear index, $\mathbf{x}_i$ a known transformation of the text token counts $\mathbf{c}_i$ and $f(\cdot)$ some link function.
- Common transformations are the identity $\mathbf{x}_i = \mathbf{c}_i$, normalization by document length $\mathbf{x}_i = \mathbf{c}_i/m_i$ with $m_i = \sum_j c_{ij}$ or just an occurence indicator $c_{ij} = \mathbb{1}(c_{ij} > 0)$.
- Choice depends on application and interpretability—identity is a reasonable default.
- L_1 penalized linear or logistic regression recommended—for simple text regression tasks seldom possible to do much better.

L_1 penalized text regression

- Lasso deviance objective function:

$$min \left\{ l(\alpha, \beta) + n\lambda \sum_{j=1}^{p} \omega_j |\beta_j| \right\}$$

- $l(\alpha, \beta)$ is an unregularized objective proportional to the negative log-likelihood log $p(\mathbf{v}_i|\mathbf{c}_i)$.
- Linear (gaussian) regression: $l(\alpha, \beta) = \sum_i (\mathbf{v}_i - \eta_i)^2$
- Logistic (binomial) regression: $l(\alpha, \beta) = -\sum_i (\eta_i \mathbf{v}_i - \log(1 + e^{\eta_i}))^2$ if $\mathbf{v}_i \in \{0, 1\}$.

L_1 penalized text regression

- Lasso deviance objective function:

$$min \left\{ l(\alpha, \boldsymbol{\beta}) + n\lambda \sum_{j=1}^{p} \omega_j |\beta_j| \right\}$$

- Common strategy to set ω_j such that the penalty cost for each coefficient is scaled by the sample standard deviation of that covariate.
- 'Rare feature upweighting' is good practice: Each covariate corresponds to a specific text token, and rare words are often most useful in differentiating between documents.
- Model selection via cross-validation, AIC, BIC.

Other text regression models

- Penalized regression most widely applied tool.
- Previously, people often just applied PCA to the term-document matrix and used some of the principal components for regression ('Latent Semantic Analysis').
- Bayesian regression methods available.
- Tree-based methods can be used, but are less useful with high-dimensional text-based inputs.
- Benefits of trees (nonlinearity, high-order interaction detection) cannot be reaped with sparse binary inputs.
- May be useful for a final prediction step after a dimension reduction from a generative model.

Generative language models

- Text regression uses counts as high-dimensional input variables, without attempting to model structure specific to language data.
- Propose a generative model for text tokens to learn about how the attributes influence word choice and account for dependencies between words and attributes.
- Define a model for $p(\mathbf{c}_i|\mathbf{v}_i)$.
- Intuitive: Causal relationship typically runs from outcomes to language. Google searches about the flu do not cause the flu.
- Supervised vs. unsupervised generative models.

Supervised generative models: Naive Bayes

- Most common supervised generative model: Naive Bayes classifier.
- Probabilistic classifier based on applying Bayes' theorem.
- Popular method for text categorization. Good baseline, relatively competitive.
- Relies on a strong (*naive*) indepencence assumption of features.
- Assume v is a class to be estimated, a univariate categorical attribute (e.g. spam).
- Probabilistic classifier: $\hat{v} = \underset{v \in V}{\operatorname{argmax}}\, p(v|d)$

Naive Bayes

- Rewrite class estimate using Bayes' rule:

$$\hat{v} = \underset{v \in V}{\operatorname{argmax}}\, p(v|d) = \underset{v \in V}{\operatorname{argmax}}\, \frac{p(d|v)p(v)}{p(d)} \ .$$

- Drop the denominator. We compute the class probability for each possible class, but $p(d)$ does not change for each class.
- Choose the class that maximizes

$$\hat{v} = \underset{v \in V}{\operatorname{argmax}}\, p(v|d) = \underset{v \in V}{\operatorname{argmax}}\, p(d|v)p(v) \ .$$

- Most probable class $\hat{v}$ has the highest product of two probabilities: the *prior probability* of the class $p(v)$, and the *likelihood* of the document $p(d|v)$.

Naive Bayes

- Without loss of generalization, we can represent a document d as a set of features $f_1, f_2, \ldots, f_p$:

$$\hat{v} = \underset{v \in V}{\operatorname{argmax}}\, p(f_1, f_2, \ldots, f_p|v)p(v) \; .$$

- Too difficult to compute directly. Make two simplifying assumptions.
- Bag-of-words: Position does not matter, features $f_1, f_2, \ldots, f_p$ only encode word identity and not position (as previously).
- Naive Bayes/conditional independence assumption: The probabilities $p(f_i|v)$ are independent given class v and can be 'naively' multiplied:

$$p(f_1, f_2, \ldots, f_p|v) = p(f_1|v) \cdot p(f_2|v) \cdot \ldots \cdot p(f_p|v)$$

Naive Bayes

- Class chosen is thus

$$\hat{v} = \underset{v \in V}{\operatorname{argmax}}\, p(v) \prod_{f \in F} p(f|v) \; .$$

- Considering only word tokens:

$$\hat{v} = \underset{v \in V}{\operatorname{argmax}}\, p(v) \prod_{c \in \mathcal{V}} p(c|v) \; .$$

- In log space to avoid underflow and increase speed:

$$\hat{v} = \underset{v \in V}{\operatorname{argmax}}\, \log p(v) + \sum_{c \in \mathcal{V}} \log p(c_j|v) \; .$$

Naive Bayes classifier: Remarks

- How can we learn $p(v)$ and $p(f_j|v)$?
- Consider using the frequencies in the data.
- Learn $p(v)$ using the share of documents in class v in the training data.

$$\hat{p}(v) = \frac{N_c}{N_{doc}}$$

- Learn $p(c_j|v)$ as the fraction of times the word c_j appears among all words in the vocabulary in all documents of class v.

$$\hat{p}(c_j|v) = \frac{count(c_j, v)}{\sum_{c \in \mathcal{V}} count(c, v)}$$

Training the Naive Bayes classifier

- Some words may not appear in a given class in the training data, e.g.

$$\hat{p}(\text{“fantastic”}|\text{positive}) = \frac{count(\text{“fantastic”}, \text{positive})}{\sum_{j=1}^{p} count(c_j, \text{positive})} = 0 \ .$$

- Naive bayes multiplies all the feature likelihoods together.
- Zero proabilitities in the likelihood term for any class will cause the probability of the class to be zero.
- Simplest solution: Laplace (add-one) smoothing.

$$\hat{p}(c_j|v) = \frac{count(c_j, v) + 1}{\sum_{c \in \mathcal{V}} count(c, v) + 1} = \frac{count(c_j, v) + 1}{\left(\sum_{c \in \mathcal{V}} count(c, v)\right) + |\mathcal{V}|}$$

Training the Naive Bayes classifier

- Ignore new words in test data that do not occur in training sample.
- Independence assumption rules out the possibility that using one token (‘hello’) influences the probability of using another (‘hi’).
- Even though this may be unrealistic, naive Bayes works suprisingly well in practice. Often used in spam filters.
- Removing stop words does not necessarily improve performance.

Unsupervised generative models

- No direct observations of the true attributes $\boldsymbol{v}_i$.
- Inference about attributes depends entirely on assumptions imposed on the structure of the model $p(\mathbf{c}_i|\mathbf{v}_i)$.
- Typical model: Each observation $\mathbf{c}_i$ is a conditionally independent draw from the vocabulary of possible tokens according to some document-specific token probability vector $\mathbf{q}_i = [q_{i1} \ldots q_{ip}]'$.
- Conditioning on document length $m_i = \sum_j c_{ij}$, this implies a multinomial distribution for the counts.

$$\mathbf{c}_i \sim \text{MN}(\mathbf{q}_i, m_i)$$

- This multinomial model underlies many contemporary statistical models for text. Popular variant: Topic model.

Topic models

- Most popular topic model is Latent Dirichlet Allocation (LDA).
- The idea is to find latent themes (*topics*) in documents.
- Essentially a clustering problem. Think of both words and documents being clustered.
- Basic idea: Every document is a mixture of topics.
  Every topic is a mixture of words.

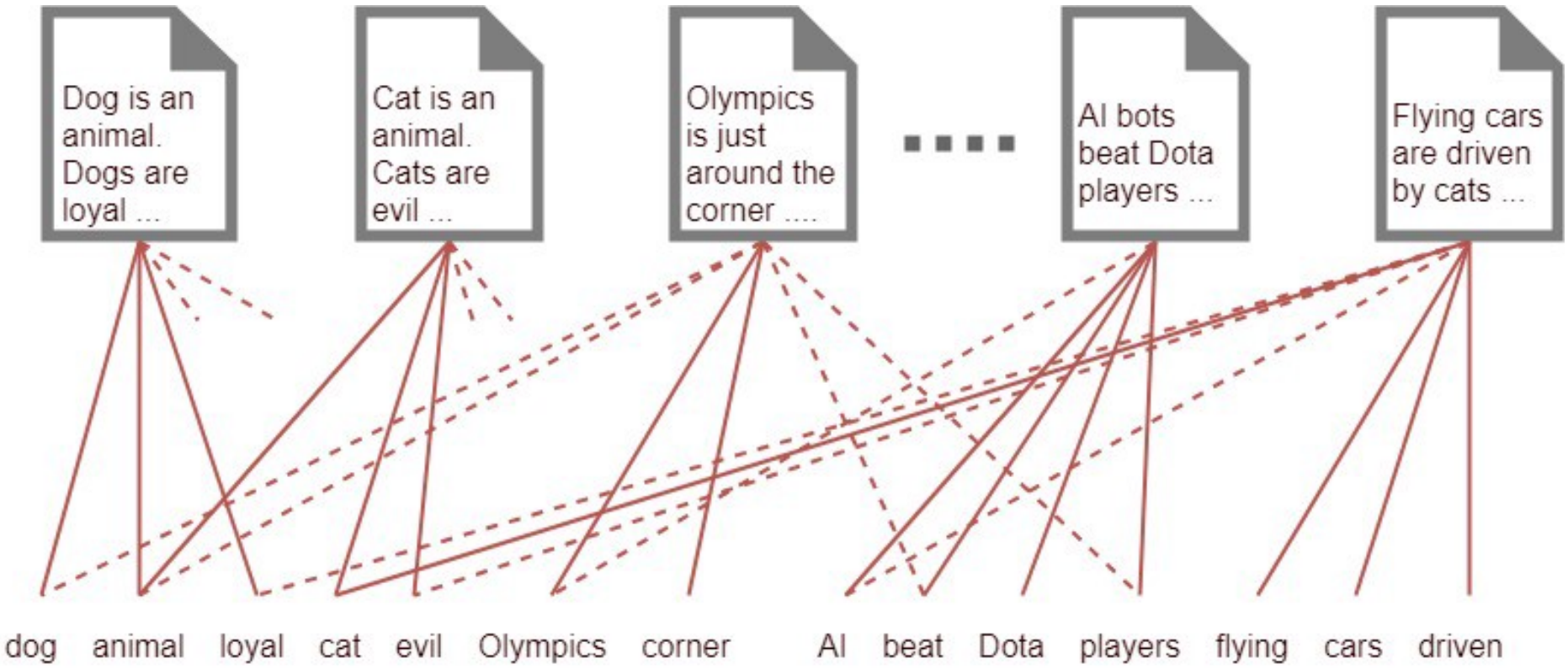
Topic models

- Topic models can be understood as factor models for the normalized token counts.
- $\mathbf{c}_i \sim \text{MN}(\mathbf{q}_i, m_i)$, link function $\mathbf{q}_i = q(\mathbf{v}_i)$ links text and attributes.
- *Topic model* specification:

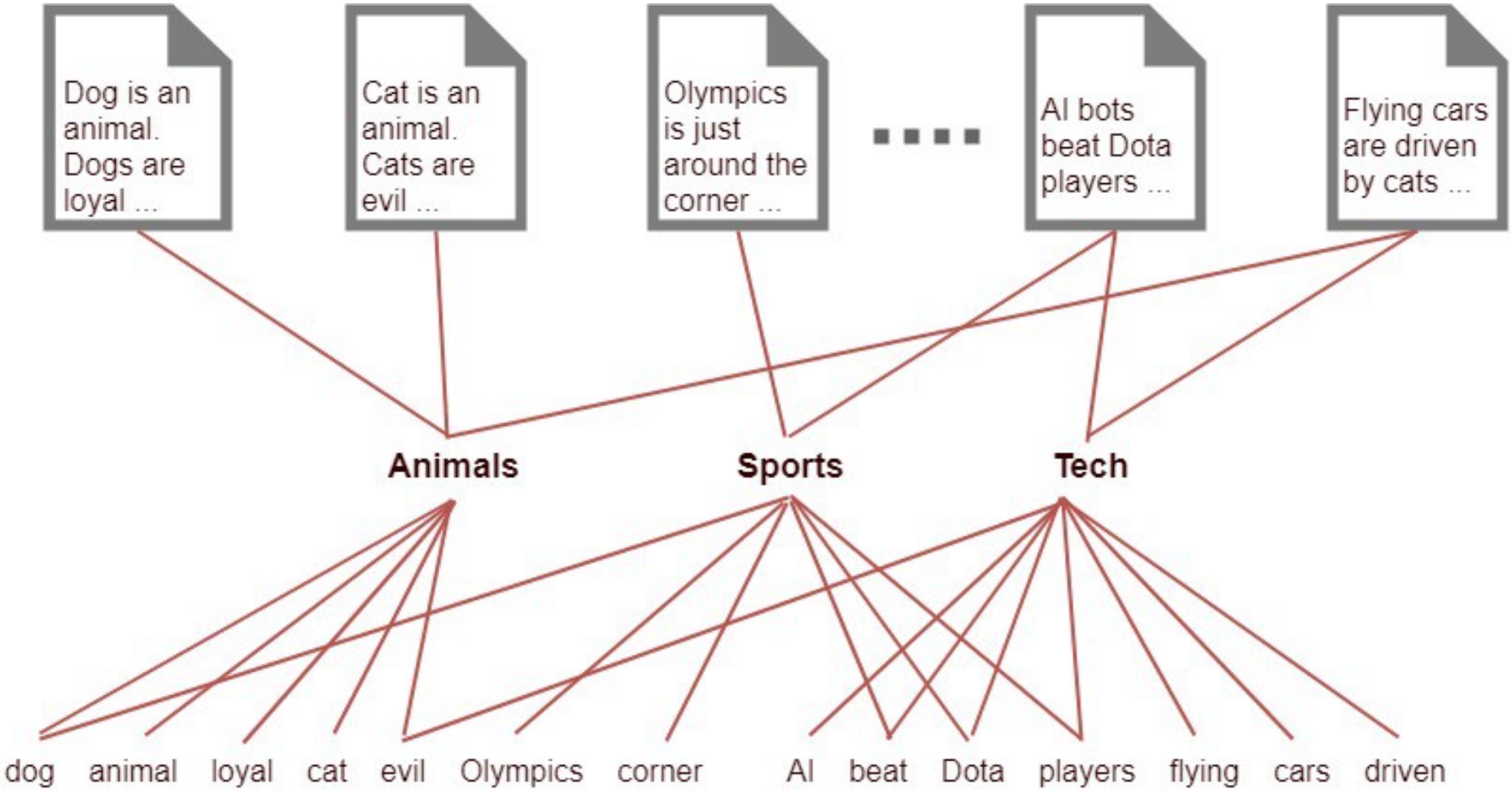
$$E\left(\frac{\mathbf{c}_i}{m_i}\right) = \mathbf{q}_i = v_{i1}\boldsymbol{\theta}_1 + v_{i2}\boldsymbol{\theta}_2 + \ldots + v_{ik}\boldsymbol{\theta}_k = \boldsymbol{\Theta}\mathbf{v}_i$$

- Each topic is a probability vector over possible tokens, $\boldsymbol{\theta}_l$, $l = 1, \ldots, k$ where $\theta_{lj} \geq 0$ and $\sum_{j=1}^{p} \theta_{lj} = 1$
- Latent attributes v_{il} are referred to as *topic weights* and restricted such that $v_{il} \geq 0$ and $\sum_{l=1}^{k} v_{il} = 1$.

Topic model


Dog is an animal. Dogs are loyal ...
Cat is an animal. Cats are evil ...
Olympics is just around the corner ....
AI bots beat Dota players ...
Flying cars are driven by cats ...
dog
animal
loyal
cat
evil
Olympics
corner
AI
beat
Dota
players
flying
cars
driven

Topic model


Dog is an animal. Dogs are loyal ...
Cat is an animal. Cats are evil ...
Olympics is just around the corner ...
AI bots beat Dota players ...
Flying cars are driven by cats ...
Animals
Sports
Tech
dog
animal
loyal
cat
evil
Olympics
corner
AI
beat
Dota
players
flying
cars
driven

Topic model/LDA

- Suppose your corpus consists of the following sentences.

```
I like to eat broccoli and bananas.
I ate a banana and spinach smoothie for breakfast.
Chinchillas and kittens are cute.
My sister adopted a kitten yesterday.
Look at this cute hamster munching on a piece of broccoli.
```

- Given this data and asked for two topics, we might find something like.

```
Sentences 1 and 2: 100% Topic A
Sentences 3 and 4: 100% Topic B
Sentence 5:        60% Topic A, 40% Topic B
Topic A: 30% broccoli, 15% bananas, 10% breakfast, 10% munching, ...
Topic B: 20% chinchillas, 20% kittens, 20% cute, 15% hamster, ...
```

➥ Interpret interpret topic A to be about food, topic B to be about animals.

LDA topic todel: Key features

- Documents exhibit multiple topics (typically relatively few).
- Probabilistic model with a corresponding generative process (each document generated by this process).
- A topic is a distribution over a fixed vocabulary. Topics are assumed to be generated first.
- Only the number of topics is specified in advance.

LDA topic todel: Generative process

To generate a document:

1. Choose a distribution over topics, i.e. a topic mixture for the document.
2. Generate each word w_i in the document by:
   - 2.1 Pick a topic from the distribution over topics.
   - 2.2 Pick a word from the corresponding topic (distribution over vocabulary).

LDA topic todel: Generative process

To generate a document:

1. Choose a distribution over topics, i.e. a topic mixture for the document.
   (Dirichlet distribution over a fixed set of k topics.)
2. Generate each word w_i in the document by:
   2.1 Pick a topic from the distribution over topics.
   (From the multinomial distribution sampled above.)
   2.2 Pick a word from the corresponding topic (distribution over vocabulary).
   (From the topic's multinomial distribution.)

- Step 1 requires a distribution over a distribution.
- Words are generated independently of other words (unigram bag-of-words model).
- Assuming this process, LDA tries to backtrack from the documents to find a set of topics that are likely to have the generated the collection.

LDA topic model: Generative process

- Notation:
  - $\boldsymbol{\theta}_{1:k}$ are the topics, each $\boldsymbol{\theta}_l$ is a distribution over the token vocabulary.
  - $\mathbf{v}_{i,1:k}$ are the weights/proportions for topic l in document i.
  - $\mathbf{q}_{i,1:p}$ are the topic assignments for word j in document i.
  - $\mathbf{c}_i$ are the observed words for document i.
- The joint distribution of hidden and observed variables:

$$p(\boldsymbol{\theta}_l, \mathbf{v}_i, \mathbf{q}_i, c_i) = \prod_{l=1}^{k} p(\boldsymbol{\theta}_l) \prod_{i=1}^{n} p(\mathbf{v}_i) \prod_{j=1}^{p} p(q_{i,j}|\theta_d) p(c_{i,j}|\theta_l, q_{i,j})$$

LDA topic model: Estimation

- $\mathbf{v}_i$ are latent, use alternating inference for $\mathbf{V}|\mathbf{\Theta}$ and $\mathbf{\Theta}|\mathbf{V}$.
- Use the expectation maximization algorithm to maximize the implied likelihood (or an equivalent Bayesian specification).
- Number of topics k is the tuning parameter.
- Choice of k often arbitrary. Daten-driven choice methods exist. Start with a sensible number, adjust to improve interpretability.
- Topic model hugely popular since its introduction (Blei 2003).
- Output from topic models can be used as input for further analysis. Researchers in political science attach political issues and beliefs to the latent topics.

Deep learning and word embedding

- Many other machine learning methods applied to text.
- Among the most common are neural networks.
- Classical neural nets tended to overfit and be difficult to tune in high-dimensional noisy settings (like text analysis).
- Recently popular ‘deep’ versions (many layers, fewer nodes per layer) work better, faster and require less tuning even in difficult problems.
- State-of-the-art, these power most of the text functions on your phone and computer, e.g. translation services or syntactic parsing.
- Rely on word embedding representations.
- Structure varies greatly across applications.

Practical considerations

- Dictionary based methods are based on prior information about the function mapping features c_i to outcomes v_i.
- Use them when such information is strong and corresponding information in the data weak.
- Example: Outcomes never observed, and the mapping of interest is not picked up by the factor structure of unsupervised methods.
- Or: Training data exists, but is too small and noisy such that a prior-driven specification of $f(\cdot)$ is deemed more reliable.

Practical considerations

- Text regression is good for predicting a single attribute, especially when a large amount of training data is available.
- Supervised generative techniques like naive Bayes or MNIR may improve prediction when p is large relative to n, but gains diminish rapidly with sample size.
- In text regression it is unwise to learn flexible functional forms unless $n \gg p$. Use linear index regression methods if this is not the case.
- Most prediction tasks with text input in social science are efficiently addressed via penalized linear regression.

Practical considerations

- If there are multiple attributes of interest and interdependence between attributes and their effect on language should be resolved, resort to generative models.
- Methods can always be combined.
- Use topic modeling for corpora of many unlabeled documents.
- Caution with the interpretation of unsupervised methods (multimodal distributions of parameter estimates). The best way to build interpretability for topic models is to add some supervision.

Model validation

- Prediction: Cross-validation, reserving a test set.
- Descriptive or causal analysis: Validate the accuracy with which the fitted model is capturing the quantity of interest.
- *Manual audits* are effective. Check some subset of fitted values against the coding a human would produce by hand.
- Especially relevant for dictionary methods: Validity hinges on particular keywords. If you have sufficient prior information to justify this, you should have sufficient information to evaluate whether the resulting classification is accurate.
- Audits can be formalized with multiple people and quantified consistency checks.
- Inspection of estimated parameters may (more likely may not) be informative.

Inferring autorship

- Ironically determining authorship mostly relies on the frequency of function (stop) words.
- First modern statistical analysis of text data, Mosteller and Wallace (1963): Who wrote the contested Federalist Papers?

- Stock and Trebbi (2003): Who invented instrumental variables?
- Was the mathematical appendix to Phillip Wright's (1928) 'The Tariff on Animal and Vegetable Oils' written by his son?

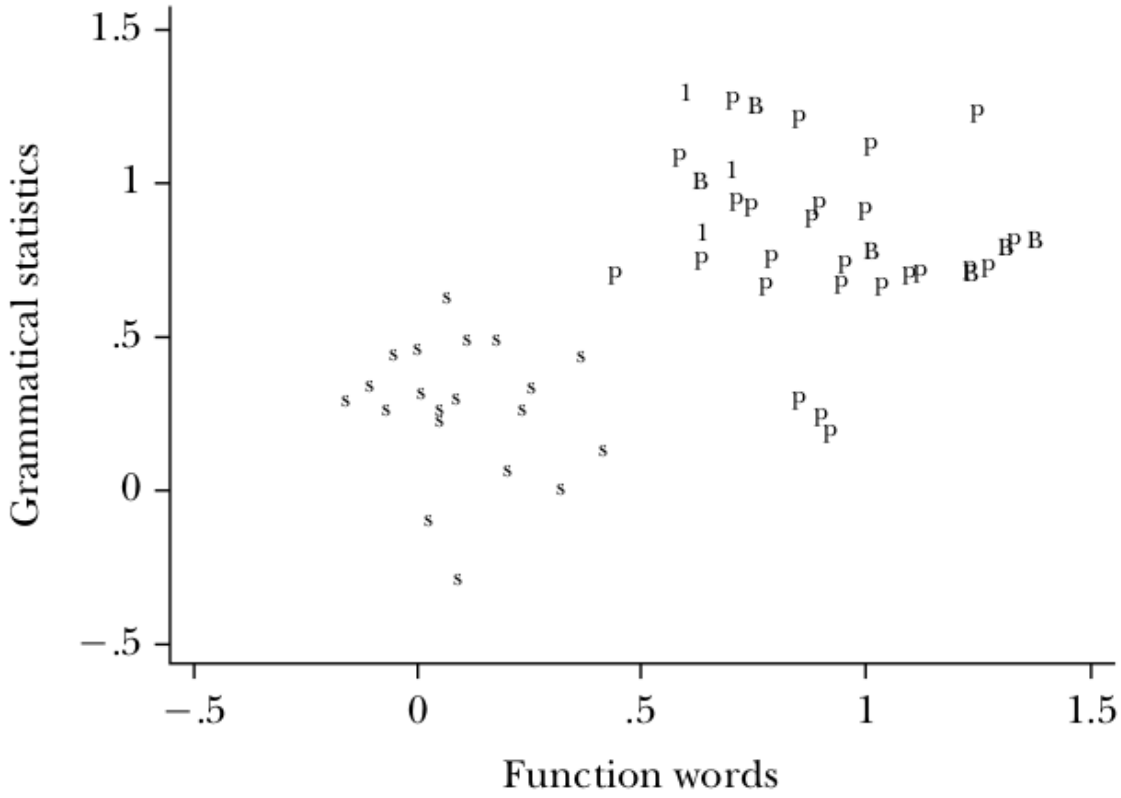
Inferring autorship

- Data features counts of function words and counts of certain grammatical constructions $c_i = (c_i^{func}, c_i^{gram})$.
- Training sample consists of 45 documents known to be written by either author.
- Test sample consists of eight blocks from the appendix and one from the main book for validation.
- Authors extract the first four principal components from $c_i = (c_i^{func}, c_i^{gram})$, then regress them on the binary authorship obtaining $(\hat{v}_i^{func}, \hat{v}_i^{gram})$.

Who invented IV?

Scatterplot of Predicted Values from Regression on First Four Principal Components: Grammatical Statistics versus Function Words

s = block undisputedly written by Sewall Wright
p = block undisputedly written by Philip G. Wright
1 = block from chapter 1, *The Tariff on Animal and Vegetable Oils*
B = block from Appendix B, *The Tariff on Animal and Vegetable Oils*



Stock prices

- Dictionary approach: Cowles (1933) categorized Wall Street Journal editorial articles as ‘bullish’, ‘bearish’ or ‘doubtful’ and used classifications to predict future Dow Jones returns.
- Many other modern applications using media announcements, Twitter messages, WSJ or investor publications.
- Mostly dictionary approaches, some text regression, few generative models.
- Related literature on central bank sentiment and announcements.

Nowcasting

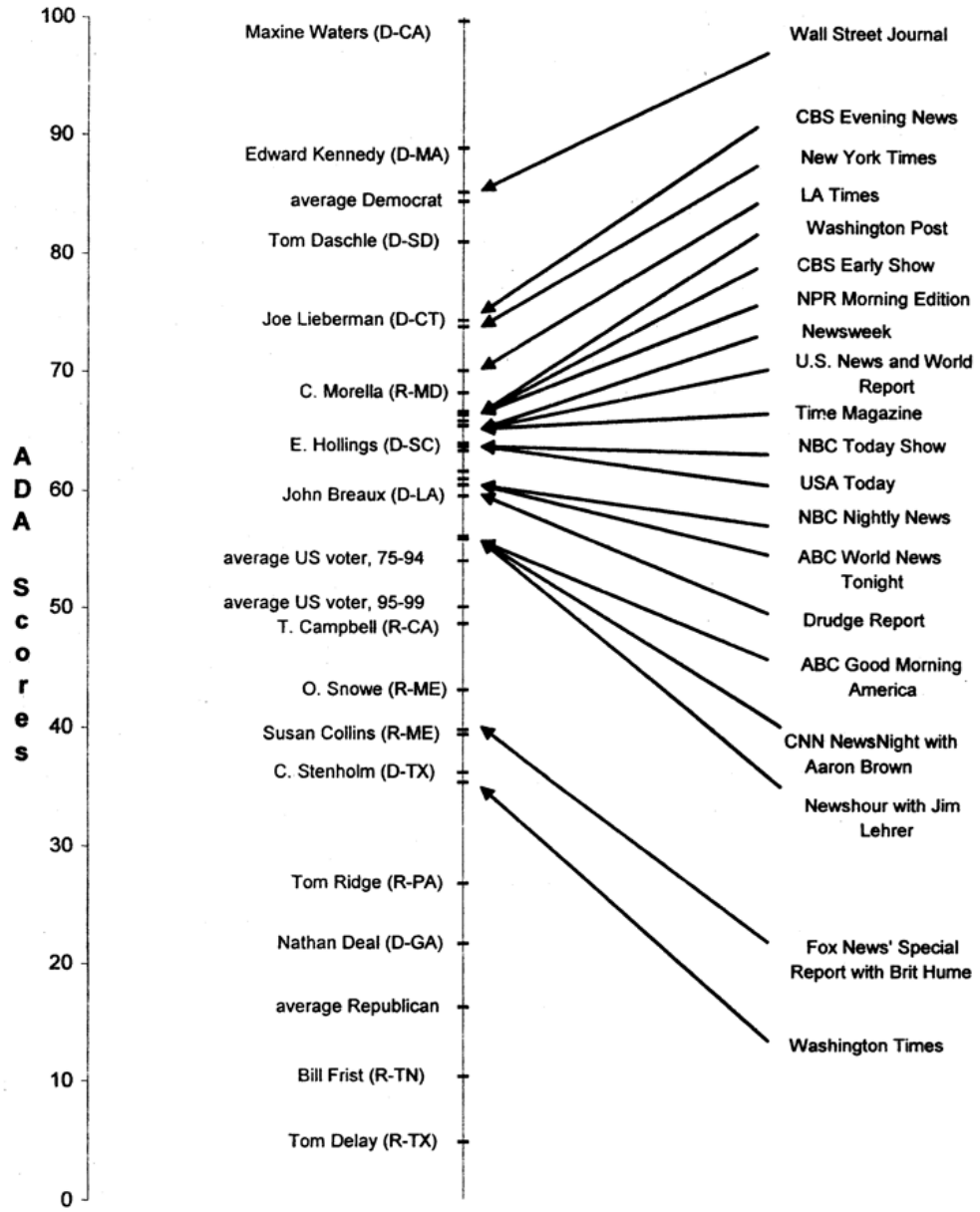
- Indicators like unemployment, retail sales and GDP are measured at low frequency and estimates are released with substantial lag.
- Others like local government corruption or racial prejudice are not captured by available measures at all.
- Text produced online can be used to construct alternative real-time estimates of the current values of these variables.
- Google Flu Trends project (Ginsberg et al. 2009), nowcasting macroeconomic variables (e.g. Choi and Varian 2012, Scott and Varian 2014, 2015), corruption in US cities (Saiz and Simonsohn 2013), racial prejudice and voting (Stephens-Davidowitz 2014).

Media Slant

- Groseclose and Milyo (2015) identify political slant of media outlets.
- Develop a model using speeches by politicians in the US congress and a left-right political ideology score from Americans for Democratic Action.
- What kind of politician does a news outlet's content sound most familiar to?
- Drastic dimension reduction by only looking at the occurence of a (supposedly) informative subset, the names of 200 think tanks.
- Use a generative model.
- Gentzkow and Shapiro (2010) do a similar analysis but omit the think tank restriction.

Media Slant

Figure 3: Distribution of Political Orientation: Media Outlets and Members of Congress



Topics in research and political debate

- Several studies apply topic models to describe how the focus of attention in specific text corpora shifts over time.
- Topics in *Science* (Blei and Lafferty 2007).
- Quinn et al. (2010) use a dynamic topic model to identify issues being discussed in the US Senate and track their relative importance over time.
- Preffered specification has 42 topics which appear coherent.

Topics being discussed in the Senate

Table 1: *Congressional Record* Topics and Key Words

Topic (Short Label)	Keys
1. Judicial Nominations	*nomine, confirm, nomin, circuit, hear, court, judg, judici, case, vacanc*
2. Constitutional	*case, court, attornei, supreme, justic, nomin, judg, m, decis, constitut*
3. Campaign Finance	*campaign, candid, elect, monei, contribut, polit, soft, ad, parti, limit*
4. Abortion	*procedur, abort, babi, thi, life, doctor, human, ban, decis, or*
5. Crime 1 [Violent]	*enforc, act, crime, gun, law, victim, violenc, abus, prevent, juvenil*
6. Child Protection	*gun, tobacco, smoke, kid, show, firearm, crime, kill, law, school*
7. Health 1 [Medical]	*diseas, cancer, research, health, prevent, patient, treatment, devic, food*
8. Social Welfare	*care, health, act, home, hospit, support, children, educ, student, nurs*
9. Education	*school, teacher, educ, student, children, test, local, learn, district, class*
10. Military 1 [Manpower]	*veteran, va, forc, militari, care, reserv, serv, men, guard, member*
11. Military 2 [Infrastructure]	*appropri, defens, forc, report, request, confer, guard, depart, fund, project*
12. Intelligence	*intellig, homeland, commiss, depart, agenc, director, secur, base, defens*

Source: Quinn et al. (2010).

Concluding remarks

- Few comprehensive textbooks on this subject.
- Field develops fast, many possible applications outside economics.
- Solid simulation evidence not always available.
- Potentially useful for economics—*how* useful depends on application.
- Penalized text regression is simple to apply.
- Latent classification methods can be illustrative. May also help obtaining measures for things which are difficult to measure otherwise.
- Possibly helpful for studies relying on unconfoundedness if data is detailed enough.
- But can be problematic: Often text is an outcome by itself rather than an exogenous covariate.