Text as Data

Big Data and Machine Learning for Applied Economics

Ignacio Sarmiento-Barbieri

Universidad de los Andes

Agenda

Text as Data: The Big Picture

- **Text is a vast source of data for research, business,etc**
- It comes connected to interesting "author" variables
  - What you buy, what you watch, your reviews
  - Group membership, who you represent, who you email
  - Market behavior, macro trends, the weather

Text as Data: Motivation

Econometrica, Vol. 78, No. 1 (January, 2010), 35–71

WHAT DRIVES MEDIA SLANT? EVIDENCE FROM U.S. DAILY NEWSPAPERS

BY MATTHEW GENTZKOW AND JESSE M. SHAPIRO[1]

We construct a new index of media slant that measures the similarity of a news outlet's language to that of a congressional Republican or Democrat. We estimate a model of newspaper demand that incorporates slant explicitly, estimate the slant that would be chosen if newspapers independently maximized their own profits, and compare these profit-maximizing points with firms' actual choices. We find that readers have an economically significant preference for like-minded news. Firms respond strongly to consumer preferences, which account for roughly 20 percent of the variation in measured slant in our sample. By contrast, the identity of a newspaper's owner explains far less of the variation in slant.

KEYWORDS: Bias, text categorization, media ownership.

Text as Data: Motivation

Gentzkow and Shapiro: What drives media slant? Evidence from U.S. daily newspapers (*Econometrica*, 2010)

- Build an economic model for newspaper demand that incorporates political partisanship (Republican vs Democrat)
  - What would be independent profit-maximizing "slant"?
  - Compare this to slant estimated from newspaper text.
- use data from Congress to isolate the phrases
- Compare phrase frequencies in the newspaper with phrase frequencies in the 2005 Congressional Record to identify whether the newspaper's language is more similar to that of a congressional Republican or a congressional Democrat

Republican	Democratic
death tax	estate tax
tax relief	tax break
personal account	private account
war on terror	war in Iraq

Text as Data: Motivation

THE JOURNAL OF FINANCE • VOL. LXII, NO. 3 • JUNE 2007

Giving Content to Investor Sentiment: The Role of Media in the Stock Market

PAUL C. TETLOCK*

ABSTRACT

I quantitatively measure the interactions between the media and the stock market using daily content from a popular *Wall Street Journal* column. I find that high media pessimism predicts downward pressure on market prices followed by a reversion to fundamentals, and unusually high or low pessimism predicts high market trading volume. These and similar results are consistent with theoretical models of noise and liquidity traders, and are inconsistent with theories of media content as a proxy for new information about fundamental asset values, as a proxy for market volatility, or as a sideshow with no relationship to asset markets.

Text as Data

- A passage in '*As You Like It*' from Shakepeare:

  All the world's a stage,
  and all the men and women merely players:
  they have their exits and their entrances;
  and one man in his time plays many parts...

Text as Data

- A passage in '*As You Like It*' from Shakepeare:

  All the world's a stage,
  and all the men and women merely players:
  they have their exits and their entrances;
  and one man in his time plays many parts...

- What the econometrian sees:

world	stage	men	women	play	exit	entrance	time
1	1	2	1	2	1	1	1

- This is the Bag-of-Words representation of text and the key is the Document Term Matrix (DTM).

Text as Data: DTM

- A document-term matrix is a mathematical matrix that describes the frequency of terms that occur in a collection of documents.
- In a document-term matrix, rows correspond to documents in the collection and columns correspond to terms.
- Example 3 documents, A, B, y C:

  1. El sol es una estrella.
  2. Un buen viajante no tiene planes.
  3. Juan tiene una mascota nueva

- First step is to find the number of unique words el, sol, es, una, estrella, un, buen, viajante, no, tiene, planes, Juan, mascota, nueva. (14 words).

Text as Data: DTM

- However, we can think that not all the words in a document have the same weight.
- El TF-IDFVectorizer (term frequency-inverse document frequency) incorporates this notion

$$TF - IDF_{ij} = tf_{ij} \times \left(\log \left(\frac{1+N}{1+df_i} \right) + 1 \right)$$

where:

- tf_{ij} is the frequency of word i in the j document
- df_{ij} number of documents that have the word i
- N number of documents

Text as Data: Text cleaning and tokenization

- However not all words are useful in the DTM, we can follow certain rules to clean these up.
- The text cleaning process, within the scope of text mining, consists of eliminating everything from the text that does not provide information about its theme, structure or content.
- There is no single way to do it, it largely depends on the purpose of the analysis and the source from which the text comes.
- Tokenizing a text consists of dividing the text into the units that make it up, ending with the simplest element with its own meaning for the analysis in question

Text as Data: Text cleaning and tokenization

Some steps include

- Convert to lowercase, drop numbers, punctuation, etc ...
  Always application specific: e.g., don't drop :-) from tweets.
- Remove a list of stop words containing irrelevant tokens .
  If, and, but, who, what, the, they, their, a, or, ...

  Be careful: one person's stopword is another's key term.
- Remove words that are super rare (in say $< \frac{1}{2}\%$, or $< 15\%$ of docs; this is application specific). For example, if Argentine occurs only once, it's useless for comparing documents.

Text as Data: Text cleaning and tokenization

Stemming and lemmatization

- For grammatical reasons, documents are going to use different forms of a word, such as *organize, organizes, and organizing.*
- Additionally, there are families of derivationally related words with similar meanings, such as *democracy, democratic, and democratization.*
- In many situations, it seems as if it would be useful for a search for one of these words to return documents that contain another word in the set.

Text as Data: Text cleaning and tokenization

Stemming and lemmatization

- The goal of both stemming and lemmatization is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form. For instance:
  - am, are, is $\Rightarrow$ be
  - car, cars, car's, cars' $\Rightarrow$ car
- The result of this mapping of text will be something like:
  - the boy's cars are different colors $\Rightarrow$
  - the boy car be differ color

Text as Data: Text cleaning and tokenization

- However, the two differ in their flavor.
- Stemming usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes.
- Lemmatization usually refers to doing things properly with the use of a vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma .
- For example:
  If confronted with the token `saw`, stemming might return just `s` whereas lemmatization would attempt to return either `see` or `saw` depending on whether the use of the token was as a verb or a noun.

Tokenization Demo

Going to use https://www.derechopenalenlared.com/libros/jardin-senderos-borges.pdf

```
## the tm library (and related plugins) is R's ecosystem for text mining.
## for an intro see http://cran.r-project.org/web/packages/tm/vignettes/tm.pdf
library(tm)

notes<-readPDF(control = list(text = "-layout -enc UTF-8"))
(elem=list(uri="jardin-senderos-borges.pdf"),  id=fname, language='es')
writeLines(content(notes)[1])
```

```
##      El jardín de los senderos que se bifurcan
##                                 Jorge Luis Borges
##
##                                                                          A Victoria Ocampo
##
## En la página 22 de la Historia de la Guerra Europea, de Liddell Hart, se lee que
## una ofensiva de trece divisiones británicas (apoyadas por mil cuatrocientas piezas
## de artillería) contra la línea Serre Montauban había sido planeada para el
## veinticuatro de julio de 1916 y debió postergarse hasta la mañana del día
## veintinueve. Las lluvias torrenciales (anota el capitán Liddell Hart) provocaron esa
## demora -nada significativa, por cierto-. La siguiente declaración, dictada, releída y
## firmada por el doctor Yu Tsun, antiguo catedrático de inglés en la Hochschule de
## Tsingtao, arroja una insospechada luz sobre el caso. Faltan las dos páginas
## iniciales.
```

Tokenization Demo

```
content(notes) <-stringi::stri_trans_general(content(notes), "Latin-ASCII")
writeLines(content(notes)[1])
```

```
##      El jardin de los senderos que se bifurcan
##                                 Jorge Luis Borges
##
##
##                                                                          A Victoria Ocampo
##
##
##
## En la pagina 22 de la Historia de la Guerra Europea, de Liddell Hart, se lee que
## una ofensiva de trece divisiones britanicas (apoyadas por mil cuatrocientas piezas
## de artilleria) contra la linea Serre Montauban habia sido planeada para el
## veinticuatro de julio de 1916 y debio postergarse hasta la manana del dia
## veintinueve. Las lluvias torrenciales (anota el capitan Liddell Hart) provocaron esa
## demora -nada significativa, por cierto-. La siguiente declaracion, dictada, releida y
## firmada por el doctor Yu Tsun, antiguo catedratico de ingles en la Hochschule de
## Tsingtao, arroja una insospechada luz sobre el caso. Faltan las dos paginas
## iniciales.

##                                          1
```

Tokenization Demo

```
docs <- Corpus(VectorSource(notes))
names(docs) <- names(notes) # no idea why this doesn't just happen
## you can then do some cleaning here
## tm_map just maps some function to every document in the corpus
docs <- tm_map(docs, content_transformer(tolower)) ## make everything lowercase
head(docs[[1]]$content)
```

```
## [1] "     el jardin de los senderos que se bifurcan\n    jorge luis borges\n\n\n  a victoria ocampo\n\n\n\n\n
en la pagina 22 de la historia de la guerra europea, de liddell hart,
se lee que\nuna ofensiva de trece divisiones britanicas (apoyadas por mil cuatrocientas piezas\nde artilleria)
contra la linea serre montauban habia sido planeada para el\nveinticuatro de julio de 1916 y debio postergarse hasta la manana del dia\nveir
por cierto-. la siguiente declaracion, dictada, releida y\nfirmada por el doctor yu tsun, antiguo catedratico de ingles
en la hochschule de\ntsingtao, arroja una insospechada luz sobre el caso.
faltan las dos paginas\niniciales.
```

Tokenization Demo

```
docs <- tm_map(docs, content_transformer(removeNumbers)) ## remove numbers
head(docs[[1]]$content)
```

```
## [1] "     el jardin de los senderos que se bifurcan\n                                                jorge luis borges\n\n\n
la pagina  de la historia de la guerra europea, de liddell hart,
se lee que\nuna ofensiva de trece divisiones britanicas (apoyadas por mil cuatrocientas piezas\nde artilleria)
contra la linea serre montauban habia sido planeada para el\nveinticuatro de julio de
y debio postergarse hasta la manana del dia\nveintinueve.
las lluvias torrenciales (anota el capitan liddell hart) provocaron esa\n
demora -nada significativa, por cierto-. la siguiente declaracion, dictada,
releida y\nfirmada por el doctor yu tsun, antiguo catedratico de ingles en la hochschule de\n
tsingtao, arroja una insospechada luz sobre el caso. faltan las dos paginas\niniciales.
```

Tokenization Demo

```
docs <- tm_map(docs, content_transformer(removePunctuation)) ## remove punctuation
head(docs[[1]]$content)
```

```
## [1] "     el jardin de los senderos que se bifurcan\n                               jorge luis borges\n\n\n
apoyadas por mil cuatrocientas piezas\nde artilleria contra la linea
serre montauban habia sido planeada para el\nveinticuatro de julio de  y
debio postergarse hasta la manana del dia\nveintinueve las lluvias torrenciales anota
el capitan liddell hart provocaron esa\ndemora nada significativa por cierto la
siguiente declaracion dictada releida y\nfirmada por el doctor yu tsun
antiguo catedratico de ingles en la hochschule de\ntsingtao arroja una insospechada
luz sobre el caso faltan las dos paginas\niniciales
```

Tokenization Demo

```
## remove stopword.  be careful with this: one's stopwords are anothers keywords.
docs <- tm_map(docs, content_transformer(removeWords), stopwords("spanish"))
head(docs[[1]]$content)
```

```
## [1] "      jardin   senderos   bifurcan\n                                  jorge luis borges\n\n\n
pagina    historia   guerra europea  liddell hart  lee \n
ofensiva  trece divisiones britanicas apoyadas  mil cuatrocientas piezas\n
artilleria   linea serre montauban habia  planeada  \n
veinticuatro  julio    debio postergarse   manana  dia\n
veintinueve  lluvias torrenciales anota  capitan liddell hart provocaron \n
demora  significativa  cierto  siguiente declaracion dictada releida \n
firmada   doctor yu tsun antiguo catedratico  ingles   hochschule \n
tsingtao arroja  insospechada luz   caso faltan  dos paginas\niniciales\n                                  \n"
```

Tokenization Demo

```
docs <- tm_map(docs, content_transformer(stripWhitespace)) ## remove excess white-spa
head(docs[[1]]$content)
```

```
## [1] " jardin senderos bifurcan jorge luis borges victoria ocampo
pagina historia guerra europea liddell hart lee ofensiva
trece divisiones britanicas apoyadas mil cuatrocientas piezas artilleria
linea serre montauban habia planeada veinticuatro julio debio postergarse
manana dia veintinueve lluvias torrenciales anota capitan liddell hart provocaron
demora significativa cierto siguiente declaracion dictada releida firmada
doctor yu tsun antiguo catedratico ingles hochschule tsingtao arroja
insospechada luz caso faltan dos paginas iniciales
```

Tokenization Demo

```
require("SnowballC")
docs <- tm_map(docs, stemDocument,language= "spanish")
head(docs[[1]]$content)
```

```
## [1] "jardin sender bifurc jorg luis borg victori ocamp pagin histori
guerr europe liddell hart lee ofens trec division britan apoy mil cuatrocient
piez artilleri line serr montaub habi plan veinticuatr juli debi posterg manan
dia veintinuev lluvi torrencial anot capit liddell hart provoc demor signific ciert
siguient declaracion dict rel firm doctor yu tsun antigu catedrat ingles hochschul
tsingta arroj insospech luz cas falt dos pagin inicial "
```

Tokenization Demo

```
## create a doc-term-matrix
dtm <- DocumentTermMatrix(docs)
dtm
```

```
## <<DocumentTermMatrix (documents: 9, terms: 1052)>>
## Non-/sparse entries: 1576/7892
## Sparsity           : 83%
## Maximal term length: 14
## Weighting          : term frequency (tf)
```

Tokenization Demo

```
dtm <- removeSparseTerms(dtm, 0.75)
dtm
```

```
## <<DocumentTermMatrix (documents: 9, terms: 118)>>
## Non-/sparse entries: 457/605
## Sparsity           : 57%
## Maximal term length: 9
## Weighting          : term frequency (tf)
```

```
## These are special sparse matrices.
class(dtm)
```

```
## [1] "DocumentTermMatrix"    "simple_triplet_matrix"
```

Tokenization Demo

```
## You can inspect them:
inspect(dtm[1:5,1:8])
```

```
## <<DocumentTermMatrix (documents: 5, terms: 8)>>
## Non-/sparse entries: 21/19
## Sparsity           : 48%
## Maximal term length: 6
## Weighting          : term frequency (tf)
## Sample             :
##     Terms
## Docs asesin bifurc capit cas despu dos espald favor
##    1      1      1     3   1     1   2      1     1
##    2      0      0     1   0     1   1      0     0
##    3      0      0     1   3     0   0      0     1
##    4      1      2     0   1     0   1      0     0
##    5      0      0     0   0     1   1      1     0
```

Tokenization Demo

```
## find words with greater than a min count
findFreqTerms(dtm,10)
```

```
##  [1] "bifurc"   "habi"     "hombr"    "jardin"   "madd"     "sender"
##  [7] "vez"      "dij"      "infinit"  "pued"     "albert"   "usted"
## [13] "laberint" "novel"    "pen"      "tiemp"    "tsui"
```

Tokenization Demo

```
## or grab words whose count correlates with given words
findAssocs(dtm, "bifurc", .9)
```

```
## $bifurc
##  sender    abri posibil
##    0.96    0.90    0.90
```

Tokenization Demo

```
tdm = TermDocumentMatrix(docs, control = list(weighting = weightTfIdf))
```

```
tdm <- removeSparseTerms(tdm, 0.75)
inspect(tdm[1:5,1:8])
```

```
## <<TermDocumentMatrix (terms: 5, documents: 8)>>
## Non-/sparse entries: 18/22
## Sparsity           : 55%
## Maximal term length: 6
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
## Sample             :
##         Docs
## Terms              1           2           3           4           5
##   asesin 0.008660997 0.000000000 0.000000000 0.006631642 0.000000000
##   bifurc 0.006393033 0.000000000 0.000000000 0.009790167 0.000000000
##   capit  0.019179098 0.004381742 0.004223556 0.000000000 0.000000000
##   cas    0.004633863 0.000000000 0.009184082 0.003548104 0.000000000
##   despu  0.008660997 0.005936189 0.000000000 0.000000000 0.006687605
##         Docs
## Terms              6 7           8
##   asesin 0.000000000 0 0.000000000
##   bifurc 0.021348996 0 0.020258442
##   capit  0.000000000 0 0.005064610
##   cas    0.003094879 0 0.007341965
##   despu  0.000000000 0 0.000000000
```

Text Regression

- Once you have text in a numeric format, we can use all the tools we learned so far
- For example: Find words more related to Republicans

$$Republican\ Share = f(X\beta) + u \tag{1}$$

- where X is the cleaned bigram DTM
- We are going to use Lasso, to select the bigrams that best predict the share of votes for republicans

Text Regression: Example (Gentzkow and Shapiro)

```
#load packages
library(textir)
#load data
data(congress109)
congress109Counts[c("Barack Obama","John Boehner"),995:998]
```

```
## 2 x 4 sparse Matrix of class "dgCMatrix"
##              stem.cel natural.ga hurricane.katrina trade.agreement
## Barack Obama        .          1                20               7
## John Boehner        .          .                14               .
```

```
congress109Ideology[1:4,1:5]
```

```
##                            name party state chamber  repshare
## Chris Cannon       Chris Cannon     R    UT       H 0.7900621
## Michael Conaway Michael Conaway     R    TX       H 0.7836028
## Spencer Bachus   Spencer Bachus     R    AL       H 0.7812933
## Mac Thornberry   Mac Thornberry     R    TX       H 0.7776520
```

Text Regression

- We are going to use Lasso, to select the bigrams that best predict the share of votes for republicans

```
f <- congress109Counts
y <- congress109Ideology$repshare
# lasso
lassoslant <- cv.gamlr(congress109Counts>0, y)
B <- coef(lassoslant$gamlr)[-1,]
head(sort(round(B[B!=0],4)),10)
```

```
##    congressional.black.caucu                family.value
##                       -0.0839                     -0.0443
##        issue.facing.american           voter.registration
##                       -0.0324                     -0.0298
##      minority.owned.business            strong.opposition
##                       -0.0284                     -0.0264
##                  civil.right        universal.health.care
##                       -0.0259                     -0.0254
## congressional.hispanic.caucu           ohio.electoral.vote
##                       -0.0187                     -0.0183
```

Text Regression

```
tail(sort(round(B[B!=0],4)),10)
```

```
##         illegal.alien        percent.growth   illegal.immigration
##                0.0079                0.0083                0.0087
##           global.war          look.forward            war.terror
##                0.0098                0.0099                0.0114
##     private.property        action.lawsuit          human.embryo
##                0.0133                0.0142                0.0226
## million.illegal.alien
##                0.0328
```

Topic Models

- Text is super high dimensional
- Some times unsupervized factor model is a popular and useful strategy with text data
- You can first fit a factor model to a giant corpus and use these factors for supervised learning on a subset of labeled documents.
- The unsupervised dimension reduction facilitates the supervised learning
- We can use PCA

PCA as a Topic Model: Example

- We have 6166 reviews, with an average length of 90 words per review, `we8there.com`.
- A useful feature of these reviews is that they contain both text and a multidimensional rating on overall experience, atmosphere, food, service, and value.
- For example, one user submitted a glowing review for Waffle House #1258 in Bossier City, Louisiana: *I normally would not revue a Waffle House but this one deserves it. The workers, Amanda, Amy, Cherry, James and J.D. were the most pleasant crew I have seen. While it was only lunch, B.L.T. and chili, it was great. The best thing was the 50' s rock and roll music, not to loud not to soft. This is a rare exception to what you all think a Waffle House is. Keep up the good work.*
  Overall: 5, Atmosphere: 5, Food: 5, Service: 5, Value: 5.

PCA as a Topic Model: Example

- After cleaning and stemming, we are left with a vocabulary of 2640 bigrams.
- For example, the first review in the document-term matrix has nonzero counts on bigrams indicating a pleasant meal at a rib joint:

```
#load packages
library(textir)
#load data
data(we8there)
x <- we8thereCounts
x[1,x[1,]!=0]
```

```
## even though larg portion  mouth water     red sauc    babi back     back rib chocol mouss
##           1            1            1            1            1            1            1
## veri satisfi
##            1
```

PCA as a Topic Model: Example

- We can apply PCA to get a factor representation of the review text.
- PC1 looks like it will be big and positive for positive reviews,

```
pca <- prcomp(x, scale=TRUE) # can take a long time

tail(sort(pca$rotation[,1]))
```

```
##     food great      staff veri      excel food high recommend      great food
##    0.007386860     0.007593374     0.007629771     0.007821171     0.008503594
##     food excel
##    0.008736181
```

- while PC4 will be big and negative

```
tail(sort(pca$rotation[,4]))
```

```
##   order got after minut  never came   ask check readi order drink order
##  0.05918712  0.05958572  0.06099509  0.06184512  0.06776281  0.07980788
```

Latent Dirichlet Allocation

- The approach of using PCA to factorize text was common before the 2000s.
- Versions of this algorithm were referred to under the label latent semantic analysis.
- However, this changed with the introduction of topic modeling, also known as Latent Dirichlet Allocation (LDA), by Blei et al. in 2003.
- These authors pointed out that the squared error loss (i.e., Gaussian model) implied by PCA is inappropriate for analysis of sparse word-count data.
- Instead, they proposed you take the bag-of-words representation seriously and model token counts as realizations from a multinomial distribution.

Latent Dirichlet Allocation

TRANSPARENCY AND DELIBERATION WITHIN THE FOMC: A COMPUTATIONAL LINGUISTICS APPROACH*

STEPHEN HANSEN
MICHAEL MCMAHON
ANDREA PRAT

How does transparency, a key feature of central bank design, affect monetary policy makers' deliberations? Theory predicts a positive discipline effect and negative conformity effect. We empirically explore these effects using a natural experiment in the Federal Open Market Committee in 1993 and computational linguistics algorithms. We first find large changes in communication patterns after transparency. We then propose a difference-in-differences approach inspired by the career concerns literature, and find evidence for both effects. Finally, we construct an influence measure that suggests the discipline effect dominates. *JEL Codes*: E52, E58, D78.

Latent Dirichlet Allocation

THE PARTICIPATION DIVIDEND OF TAXATION: HOW CITIZENS IN CONGO ENGAGE MORE WITH THE STATE WHEN IT TRIES TO TAX THEM*

JONATHAN L. WEIGEL

This article provides evidence from a fragile state that citizens demand more of a voice in the government when it tries to tax them. I examine a field experiment randomizing property tax collection across 356 neighborhoods of a large Congolese city. The tax campaign was the first time most citizens had been registered by the state or asked to pay formal taxes. It raised property tax compliance from 0.1% in control to 11.6% in treatment. It also increased political participation by about 5 percentage points (31%): citizens in taxed neighborhoods were more likely to attend town hall meetings hosted by the government or submit evaluations of its performance. To participate in these ways, the average citizen incurred costs equal to their daily household income, and treated citizens spent 43% more than control. Treated citizens also positively updated about the provincial government, perceiving more revenue, less leakage, and a greater responsibility to provide public goods. The results suggest that broadening the tax base has a "participation dividend," a key idea in historical accounts of the emergence of inclusive governance in early modern Europe and a common justification for donor support of tax programs in weak states. *JEL* Codes: H20, P48, D73.

Latent Dirichlet Allocation

TABLE VII

TOPICS OF CITIZEN COMMENTS AT TOWN HALLS AND WRITTEN-IN COMMENTS ON SUBMITTED EVALUATIONS

Order	(1)	(2)	(3)	(4)	(5)
Panel A: Topics of citizen comments at town hall meetings					
1	pay	tax	necessary	pay	pay
2	necessary	population	population	take	must
3	population	necessary	collectors	without	population
4	tax	pay	pay	decision	why
5	why	know	know	why	others
6	agents	do	see	necessary	collectors
7	time	collectors	tax	participation	agents
8	collectors	why	without	tax	nothing
9	communes	nothing	information campaign	others	participation
10	manager	schools	transparency	agents	tax
Panel B: Topics of written-in comments on submitted evaluations					
1	government	government	government	government	government
2	water	provincial	provincial	provincial	province
3	ask	should	should	work	country
4	roads	more	population	province	leaders
5	electricity	work	especially	do	population
6	improve	public	erosion	better	good
7	jobs	goods	needs	ask	ask
8	people	concerning	people	would	development
9	more	ask	security	central	love
10	who	because	take	Kasaï	could

Notes. This table reports the first ten words in each of the five main topics identified by latent Dirichlet allocation (Blei, Ng, and Jordan 2003) applied to two sources of text that offer insight into citizens' reasons

Latent Dirichlet Allocation

- Blei et al. proposed you take the bag-of-words representation seriously and model token counts as realizations from a multinomial distribution.
- Topic models are built on a simple document generation process:
  - For each word, pick a "topic" k. This topic is defined through a probability vector over words, say, θ_k with probability θ_{kj} for each word j.
  - Then draw the word according to the probabilities encoded in θ_k .
- After doing this over and over for each word in the document, you have proportion ω_{i1} from topic 1, ω_{i2} from topic 2, and so on.

Latent Dirichlet Allocation

- This basic generation process implies that the full vector of word counts, x_i, has a multinomial distribution:

$$x_i \sim MN(\omega_{i1}\theta_1 + \cdots + \omega_{iK}\theta_K, m_i) \tag{2}$$

- where $m_i = \sum_j x_{ij}$ is the total document length and, for example,
- the probability of word j in document i will be $\sum_k \omega_{ik}\theta_{kj}$

Latent Dirichlet Allocation vs PCA

- Recall our PC model:

$$E(x_i) = \delta_{i1} f_1 + \cdots + \delta_{iK} f_K \tag{3}$$

- The analogous topic model representation, implied by the above equation, is

$$E(\frac{x_i}{m_i}) = \omega_{i1} \theta_1 + \cdots + \omega_{iK} \theta_K \tag{4}$$

- such that topic score ω_{ik} is like PC score δ_{ik} and
- θ_k topic probabilities are like rotations f_k.
- The distinction is that the multinomial in implies a different loss function (from a multinomial) rather than the sums of squared errors that PCA minimizes.
- Note that we condition on document length here so that topics are driven by relative rather than absolute term usage.

LDA: Example

```
library(textir)

library(maptpx) # for the topics function

data(we8there)

# you need to convert from a Matrix to a `slam' simple_triplet_matrix
x <- as.simple_triplet_matrix(we8thereCounts)

# to fit, just give it the counts, number of `topics' K, and any other args
tpc <- topics(x,K=10)
```

```
##
## Estimating on a 6166 document collection.
## Fitting the 10 topic model.
## log posterior increase: 4441.8, 461.4, 101.5, 57.4, 51, 19.2, 26.2, 15.3, 15.4, 11.7, 6.7, 12.2, 8, 10.1,
4.8, 5.3, 3.2, 6.6, 2.8, 7, 3.6, 3.9, 6.7, 5.5, 8.6, 5, 11, 10.3, 12, 7.9, 12.1, 9, 8.8, 13.9, 8.6, 7.3, 6.1,
4.9, 4.3, 12, 11.1, 8.7, 3.2, 2.8, 5.1, 1.9, 2.6, 2.4, 4.9, 2.9, 1.5, 2.5, 4.7, 1.7, 0.9, 1.4, 0.7, 2.5, 2.2,
1.7, 1, 1.3, 1.5, 2, 0.8, 1.7, 0.5, 0.2, 0.5, 0.6, 0.9, 3.9, 0.5, 0.6, 0.4, 0.2, 0.8, 0.2, 1.4, 0.3, 0.5, 0.6, done.
```

LDA: Example

- Choosing the number of topics

```
# If you supply a vector of topic sizes, it uses a Bayes factor to choose
# (BF is like exp(-BIC), so you choose the bigggest BF)
# the algo stops if BF drops twice in a row
tpcs <- topics(x,K=5*(1:5), verb=1) # it chooses 10 topics
```

```
##
## Estimating on a 6166 document collection.
## Fit and Bayes Factor Estimation for K = 5 ... 25
## log posterior increase: 2853.9, 327.1, 85.3, 36.7, 25.9, 19.9, 13.8, 11.6, 9.6, 11.4, 20.3, 7.1, ..., done.
## log BF( 5 ) = 79521.94
## log posterior increase: 4626.7, 197.4, 53, 24.9, 19, 9.3, 7.4, 4.6, 5.2, 3.4, 2.3, 1.7, 0.8, ..., done.
## log BF( 10 ) = 87157.28
## log posterior increase: 3445, 170.2, 49.8, 23.6, 14.1, 31.4, 16.2, 4.8, 6.6, 5.5, 1.9, 5.9, ..., done.
## log BF( 15 ) = 3334.33
## log posterior increase: 2327.1, 139.8, 39.5, 16.7, 20.1, 5.3, 4.5, 3, 3.4, 2.9, 4.4, 1.8, ..., done.
## log BF( 20 ) = -66254.44
```

Topic Models: Example

- Interpretation

```
# summary prints the top `n' words for each topic,
# under ordering by `topic over aggregate' lift:
#    the topic word prob over marginal word prob.
summary(tpcs, n=10)
```

```
##
## Top 10 phrases by topic-over-null term lift (and usage %):
##
## [1] 'food great', 'great food', 'great servic', 'veri good', 'food veri', ... (14.6)
## [2] 'high recommend', 'italian food', 'best italian', 'mexican food', ...  (11.6)
## [3] 'over minut', 'never go', 'go back', 'flag down', 'anoth minut', ...  (10.4)
## [4] 'enough share', 'open daili', 'highlight menu', 'until pm', ...  (10.4)
## [5] 'never return', 'one worst', 'don wast', 'wast time', ...  (9.4)
## [6] 'good work', 'best kept', 'out world', 'great experi', ... (9.1)
## [7] 'thai food', 'veri pleasant', 'ice cream', 'breakfast lunch', ...  (9)
## [8] 'take out', 'best bbq', 'can get', 'pork sandwich', 'home cook', ...  (9)
## [9] 'food good', 'food place', 'chees steak', 'good select', 'food pretti',...  (8.7)
## [10] 'wasn whole', 'came chip', 'got littl', 'over drink', 'took seat',...  (7.8)
##
## Log Bayes factor and estimated dispersion, by number of topics:
##
##              5       10      15        20
## logBF 79521.94 87157.28 3334.33 -66254.44
## Disp      7.09     4.96    3.95      3.33
##
## Selected the K = 10 topic model
```

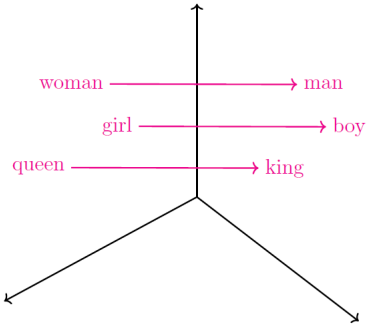
Word Embeddings

Word Embedding

- This is a “new” method that have come out of work in deep learning.
- Word embedding was originally motivated as a technique for dimension reduction on the inputs to a deep neural network.
- However, it imposes a spatial structure on words,
  - Allowing to get meanings from distance among words
  - Consider the algebra behind combinations of words in documents.

Word Embedding

- In the original deep learning context, embedding layers replace each word with a vector value
  - for example, king becomes the location [1,0.5, 0.25] in a three-dimensional embedding space



Word Embedding

- Word embeddings preserve semantic relationships.
  - Words with similar meaning have similar representations.
  - Dimensions induced by word differences can be used to identify cultural concepts
- Compare this to the standard bag-of-words representation, where hotdog would be represented as a binary vector that is as long as there are words in the vocabulary, say, p.
- This binary vector will have p–1 zeros and a one in the *king* dimension.
- The word embedding has translated the language representation from a large binary space to a smaller real-valued (and much richer) space.

Word Embedding

- There are a variety of different embedding algorithms—as many as there are different architectures for deep neural networks.
- The most common and general embeddings are built around word co-occurrence matrices.
- This includes the popular `Glove` and `Word2Vec` frameworks.
- What is co-occurrence?
  - Two words co-occur if they appear within the same sentence and within b words of each other. Where b is the "window size"
  - For a vocabulary size p, this leads to a sparse p × p co-occurrence matrix where each [i, j] entry is the number of times that words i and j co-occur. Call this matrix C.
  - A word embedding algorithm seeks to approximate C as the product of two lower-dimensional matrices

Word Embedding

- A word embedding algorithm seeks to approximate C as the product of two lower-dimensional matrices

$$C \approx UV' \tag{5}$$

- Here, U and V are each $p \times K$ dimensional dense and real valued matrices.
- K is the dimension of the embedding space; hence, $K << p$ and both U and V are very tall and thin matrices.
- Each row of U and of V, u_j and v_j is then a K-dimensional embedding of the jth word.
- The implication is that these embeddings summarize the meaning of words as their inner product defines how much you expect them to co-occur.

  Recall that the inner product is a standard measure of distance in linear algebra (e.g. $e'e$)

Word Embedding

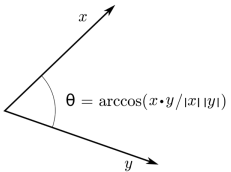
- One way to find U and V is to solve $C \approx UV'$ through the singular value decomposition (SVD).
- These locations were originally viewed as an intermediate output—as a processing step for inputs to a deep neural network.
- However, social scientists and linguists have discovered that the space of word locations contains rich information about the language of the documents used to train the embedding.

Word Embedding

- Word embeddings preserve semantic relationships.
  - Words with similar meaning have similar representations.
  - Dimensions induced by word differences can be used to identify cultural concepts.
- For example, the vector difference `man - woman` isolates a gender dimension in the space.
- The dimensions are useful because they produce quantitative measures of similarity between the associated concepts and specific words in the corpus.

Word Embedding

- In this case, we can understand the gender connotation of a given word by taking the cosine of the angle between the vector representation of the word and the differenced vector representing the gender dimension
- This is because the cosine of the angle, can be interpreted as a similarity measure.

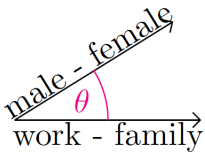


- The similarity ranges from -1 meaning exactly opposite, to 1 meaning exactly the same, with 0 indicating orthogonality or decorrelation, while in-between values indicate intermediate similarity or dissimilarity.

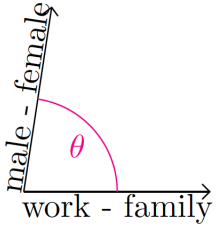
Word Embedding

Figure 2: Measuring Gender Stereotypes using Cosine Similarity

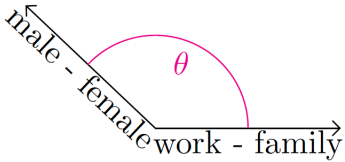
(a) Slant ≈ 1



(b) Slant ≈ 0



(c) Slant ≈ -1



Word Embedding

- Words with male connotations – e.g. male first names – are going to be positively correlated with `man - woman`.
- Female words, in turn, will be negatively correlated with the dimension.
- This framework provides an intuitive approach to measuring stereotypical associations in a given corpus.
- Bolukbasi et al (2016) is a nice example

Word Embedding: Example 1

Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings

Tolga Bolukbasi[1], **Kai-Wei Chang**[2], **James Zou**[2], **Venkatesh Saligrama**[1,2], **Adam Kalai**[2]
[1]Boston University, 8 Saint Mary's Street, Boston, MA
[2]Microsoft Research New England, 1 Memorial Drive, Cambridge, MA
tolgab@bu.edu, kw@kwchang.net, jamesyzou@gmail.com, srv@bu.edu, adam.kalai@microsoft.com

Abstract

The blind application of machine learning runs the risk of amplifying biases present in data. Such a danger is facing us with *word embedding*, a popular framework to represent text data as vectors which has been used in many machine learning and natural language processing tasks. We show that even word embeddings trained on Google News articles exhibit female/male gender stereotypes to a disturbing extent. This raises concerns because their widespread use, as we describe, often tends to amplify these biases. Geometrically, gender bias is first shown to be captured by a direction in the word embedding. Second, gender neutral words are shown to be linearly separable from gender definition words in the word embedding. Using these properties, we provide a methodology for modifying an embedding to remove gender stereotypes, such as the association between the words *receptionist* and *female*, while maintaining desired associations such as between the words *queen* and *female*. Using crowd-worker evaluation as well as standard benchmarks, we empirically demonstrate that our algorithms significantly reduce gender bias in embeddings while preserving the its useful properties such as the ability to cluster related concepts and to solve analogy tasks. The resulting embeddings can be used in applications without amplifying gender bias.

Word Embedding: Example 1

- They trained a standard word2vec embedding algorithm on the Google News corpora of news articles.
- Then look at the differences between established gender words (for example, the vector for man minus the vector for woman, or father minus mother) to establish an axis in the embedding space that spans from masculinity to femininity.
- They then calculate the location along this axis for a large number of terms that should be gender-neutral.
- The embedding space has learned—from how the words are used in news articles—that these professions are stereotypically viewed as female and male occupations.

Word Embedding: Example 1

Extreme *she*	Extreme *he*
1. homemaker	1. maestro
2. nurse	2. skipper
3. receptionist	3. protege
4. librarian	4. philosopher
5. socialite	5. captain
6. hairdresser	6. architect
7. nanny	7. financier
8. bookkeeper	8. warrior
9. stylist	9. broadcaster
10. housekeeper	10. magician

Gender stereotype *she-he* analogies

sewing-carpentry	registered nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	lovely-brilliant

Gender appropriate *she-he* analogies

queen-king	sister-brother	mother-father
waitress-waiter	ovarian cancer-prostate cancer	convent-monastery

Figure 1: **Left** The most extreme occupations as projected on to the *she*−*he* gender direction on w2vNEWS. Occupations such as *businesswoman*, where gender is suggested by the orthography, were excluded. **Right** Automatically generated analogies for the pair *she-he* using the procedure described in text. Each automatically generated analogy is evaluated by 10 crowd-workers to whether or not it reflects gender stereotype.

Word Embedding: Example 2

Language from police body camera footage shows racial disparities in officer respect

Rob Voigt[a,1], Nicholas P. Camp[b], Vinodkumar Prabhakaran[c], William L. Hamilton[c], Rebecca C. Hetey[b], Camilla M. Griffiths[b], David Jurgens[c], Dan Jurafsky[a,c], and Jennifer L. Eberhardt[b,1]

[a]Department of Linguistics, Stanford University, Stanford, CA 94305; [b]Department of Psychology, Stanford University, Stanford, CA 94305; and [c]Department of Computer Science, Stanford University, Stanford, CA 94305

Contributed by Jennifer L. Eberhardt, March 26, 2017 (sent for review February 14, 2017; reviewed by James Pennebaker and Tom Tyler)

Using footage from body-worn cameras, we analyze the respectfulness of police officer language toward white and black community members during routine traffic stops. We develop computational linguistic methods that extract levels of respect automatically from transcripts, informed by a thin-slicing study of participant ratings of officer utterances. We find that officers speak with consistently less respect toward black versus white community members, even after controlling for the race of the officer, the severity of the infraction, the location of the stop, and the outcome of the stop. Such disparities in common, everyday interactions between police and the communities they serve have important implications for procedural justice and the building of police–community trust.

racial disparities | natural language processing | procedural justice | traffic stops | policing

some have argued that racial disparities in perceived treatment during routine encounters help fuel the mistrust of police in the controversial officer-involved shootings that have received such great attention. However, do officers treat white community members with a greater degree of respect than they afford to blacks?

We address this question by analyzing officers' language during vehicle stops of white and black community members. Although many factors may shape these interactions, an officer's words are undoubtedly critical: Through them, the officer can communicate respect and understanding of a citizen's perspective, or contempt and disregard for their voice. Furthermore, the language of those in positions of institutional power (police officers, judges, work superiors) has greater influence over the course of the interaction than the language used by those with less power (12–16). Measuring officer language thus provides a quantitative lens on one key aspect of the quality or tone of

Word Embedding: Example 3

Stereotypes in High-Stakes Decisions:

Evidence from U.S. Circuit Courts

Elliott Ash, ETH Zurich

Daniel L. Chen, Toulouse School of Economics

Arianna Ornaghi, University of Warwick*

March 12, 2020

Abstract

Stereotypes are thought to be an important determinant of decision making, but they are hard to systematically measure, especially for individuals in policy-making roles. In this paper, we propose and implement a novel language-based measure of gender stereotypes for the high-stakes context of U.S. Appellate Courts. We construct a judge-specific measure of gender-stereotyped language use – *gender slant* – by looking at the linguistic association of words identifying gender (male versus female) and words identifying gender stereotypes (career versus family) in the judge's authored opinions. Exploiting quasi-random assignment of judges to cases and conditioning on detailed biographical characteristics of judges, we study how gender stereotypes influence judicial behavior. We find that judges with higher slant vote more conservatively on women's rights' issues (e.g. reproductive rights, sexual harassment, and gender discrimination). These more slanted judges also influence workplace outcomes for female colleagues: they are less likely to assign opinions to female judges, they are more likely to reverse lower-court decisions if the lower-court judge is a woman, and they cite fewer female-authored opinions.

Word Embedding: Demo

```
library(text2vec)
load('shakes_words_df_4text2vec.RData')
head(shakes_words)
```

```
##                   id         word
## 1 A_Lover_s_Complaint          nor
## 2 A_Lover_s_Complaint        gives
## 3 A_Lover_s_Complaint           it
## 4 A_Lover_s_Complaint satisfaction
## 5 A_Lover_s_Complaint           to
```

Word Embedding: Demo

```
shakes_words_ls <- list(shakes_words$word)
it <- itoken(shakes_words_ls, progressbar = FALSE)
shakes_vocab <- create_vocabulary(it)
head(shakes_vocab)
```

```
## Number of docs: 1
## 0 stopwords:  ...
## ngram_min = 1; ngram_max = 1
## Vocabulary:
##    term term_count doc_count
## 1:    1          1         1
## 2:  100          1         1
## 3:  105          1         1
## 4:  110          1         1
## 5:  115          1         1
## 6:  120          1         1
```

Word Embedding: Demo

```
shakes_vocab <- prune_vocabulary(shakes_vocab, term_count_min= 5)
head(shakes_vocab)
```

```
## Number of docs: 1
## 0 stopwords:  ...
## ngram_min = 1; ngram_max = 1
## Vocabulary:
##         term term_count doc_count
## 1:    abbess          5         1
## 2: abilities          5         1
## 3: accessary          5         1
## 4:       ace          5         1
## 5:    adders          5         1
## 6:  adjudged          5         1
```

Word Embedding: Demo

- The next step is to create the token co-occurrence matrix (TCM).
- The definition of whether two words occur together is arbitrary.

```
# maps words to indices
vectorizer <- vocab_vectorizer(shakes_vocab)

# use window of 10 for context words
shakes_tcm <- create_tcm(it, vectorizer, skip_grams_window = 10)
```

```
## [1] "dgTMatrix"
## attr(,"package")
## [1] "Matrix"
```

Word Embedding: Demo

```
glove <- GlobalVectors$new(rank = 50, x_max = 10)
shakes_wv_main = glove$fit_transform(shakes_tcm, n_iter = 10, convergence_tol = 0.01, n
```

```
##                [,1]       [,2]       [,3]       [,4]        [,5]        [,6]
## abbess     0.2964265 -0.4814041 -0.5277942  0.1603451 -0.33288023 -0.06167469
## abilities 0.5948034 -0.2549422 -0.4959693 -0.1469056  0.02951588  0.06263081
##                [,7]       [,8]       [,9]      [,10]       [,11]       [,12]
## abbess    -0.2900518 0.64890759 0.2904531 -0.08035999 -0.24630010 -0.09005264
## abilities -0.1954147 0.03472178 0.2271042  0.39861965  0.02489009 -0.32430383
##               [,13]      [,14]       [,15]      [,16]       [,17]       [,18]
## abbess    -0.1990652 -0.2910604  0.04595361 -0.5171293 -0.45380541 -0.4439173
## abilities -0.3840877 -0.3498018 -0.02203621  0.3182352  0.08673719  0.5595798
##               [,19]       [,20]       [,21]      [,22]       [,23]       [,24]
## abbess    -0.2514847 -0.169213265 -0.14375199 -0.1447455 -0.0290927 -0.3149967
## abilities -0.2777749 -0.002271206 -0.01420341  0.1749662  0.2344822  0.4177491
##               [,25]      [,26]      [,27]      [,28]      [,29]       [,30]
## abbess    -0.02706715 0.2590832 0.1625847 0.3466278 0.3888646  0.02496528
## abilities  0.05953148 0.3028767 0.3675219 0.2704012 0.1721741 -0.05913714
##               [,31]      [,32]      [,33]      [,34]       [,35]      [,36]
## abbess    -0.30768450  0.2165329 -0.1542654  0.1057229 -0.07166596 0.07947303
## abilities  0.01580654 -0.2655355  0.4817880 -0.1666721  0.20673927 0.12738403
##               [,37]      [,38]       [,39]       [,40]       [,41]
## abbess     0.06595267 -0.4524213 -0.004784943  0.04437362 -0.04464955
## abilities -0.25639891  0.2497544  0.453300126 -0.30099353 -0.32381240
##               [,42]      [,43]      [,44]       [,45]       [,46]      [,47]
## abbess    -0.1533143 0.22281612 -0.2276379  0.07420851 -0.02704191 0.2421737
## abilities -0.3664191 0.07905839  0.3145302 -0.28642598  0.15343415 0.2737963
##               [,48]      [,49]       [,50]
## abbess    -0.4376982 0.5931364  0.150960574
## abilities  0.1293828 0.1603056 -0.009560369
```

Word Embedding: Demo

```
dim(shakes_wv_main)
```

```
## [1] 9094   50
```

```
shakes_wv_context <- glove$components

dim(shakes_wv_context)
```

```
## [1]   50 9094
```

```
# Either word-vectors matrices could work, but the developers of the technique
# suggest the sum/mean may work better
shakes_word_vectors <- shakes_wv_main + t(shakes_wv_context)
```

Word Embedding: Demo

- We can play now

```
rom <- shakes_word_vectors["romeo", , drop = F]

cos_sim_rom <- sim2(x =shakes_word_vectors, y = rom, method = "cosine", norm = "l2")
```

```
##     romeo    juliet    tybalt     nurse  benvolio  banished
## 1.0000000 0.7712391 0.7575977 0.6697068 0.6517349 0.6436404
```

Word Embedding: Demo

```
love <- shakes_word_vectors["love", , drop = F]
```

```
cos_sim_rom <- sim2(x <- shakes_word_vectors, y = love, method = "cosine", norm = "l2
head(sort(cos_sim_rom[,1], decreasing <- T), 10)
```

```
##      love      that       but       not        my        so     heart       for
## 1.0000000 0.8267632 0.8166838 0.8119636 0.8015684 0.8005663 0.7894302 0.7865893
##       yet         i
## 0.7849654 0.7837292
```

Word Embedding: Demo

```
test <- shakes_word_vectors["romeo", , drop = F] -
  shakes_word_vectors["mercutio", , drop = F] +
  shakes_word_vectors["nurse", , drop = F]

cos_sim_test <- sim2(x = shakes_word_vectors, y = test, method = "cosine", norm = "l2"
head(sort(cos_sim_test[,1], decreasing = T), 10)
```

```
##     nurse    juliet     romeo      lady    mother       bed         o      wife
## 0.8904362 0.7584004 0.7179267 0.6440354 0.6374490 0.5880860 0.5756074 0.5638571
##   capulet    dromio
## 0.5520459 0.5507196
```

Word Embedding: Demo

```
test <- shakes_word_vectors["romeo", , drop = F] -
  shakes_word_vectors["juliet", , drop = F] +
  shakes_word_vectors["cleopatra", , drop = F]

cos_sim_test <- sim2(x = shakes_word_vectors, y = test, method = "cosine", norm = "l2"
head(sort(cos_sim_test[,1], decreasing = T), 3)
```

```
## cleopatra    antony    caesar
## 0.8630470 0.7011640 0.6477585
```

Further Readings

- Bolukbasi, T., Chang, K. W., Zou, J. Y., Saligrama, V., & Kalai, A. T. (2016). Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In Advances in neural information processing systems (pp. 4349-4357).
- Clark, M (2018). An Introduction to Text Processing and Analysis with R. https://m-clark.github.io/text-analysis-with-R/ Rstudio (2020). Tutorial TensorFlow https://tensorflow.rstudio.com/tutorials/beginners/basic-ml/tutorial_basic_classification/
- Taddy, M. (2019). Business data science: Combining machine learning and economics to optimize, automate, and accelerate business decisions. McGraw Hill Professional.
- Voigt, R., Camp, N. P., Prabhakaran, V., Hamilton, W. L., Hetey, R. C., Griffiths, C. M., ... & Eberhardt, J. L. (2017). Language from police body camera footage shows racial disparities in officer respect. Proceedings of the National Academy of Sciences, 114(25), 6521-6526.