Lecture 27: Text as Data

Big Data and Machine Learning for Applied Economics
Econ 4676

Ignacio Sarmiento-Barbieri

Universidad de los Andes

November 17, 2020

Announcements

- Problem Set 3: Check the data set, I changed it. I had forgotten a variable `Npobres`. You can still use the previous one to train the model. But the submission should be with the new data set.

- The submission of the .csv is on Wednesday at November 18 at 8:00 pm. Please send it via slack with the number of parameters in your model. If you forget to send me the number of parameters I'll assign $100,000$

- We need to decide the Final Exam Window: I posted a poll (please vote once). Possible dates:
  - December 3 from 2pm (Thursday) to December 5, 2pm (Saturday)
  - December 7 from 8am (Monday) to December 9, 8am (Wednesday)
  - December 9 from 8am (Wednesday) to December, 11 8am (Friday)
  - If none of these work we can tweak them a little bit
- Exam dates are from December 7 to 17
- Project proposal deadline December 7

Agenda

1. Recap: XGBoost
2. Text as Data
   - Tokenization
   - Tokenization Demo
3. Text Regression
   - Text Regression: Example
4. Topic Models
   - PCA
5. Review & Next Steps
6. Further Readings

XGBoost: Demo

- From Caret's manual
- eXtreme Gradient Boosting
  - method = 'xgbTree'
  - Type: Regression, Classification
  - Tuning parameters:

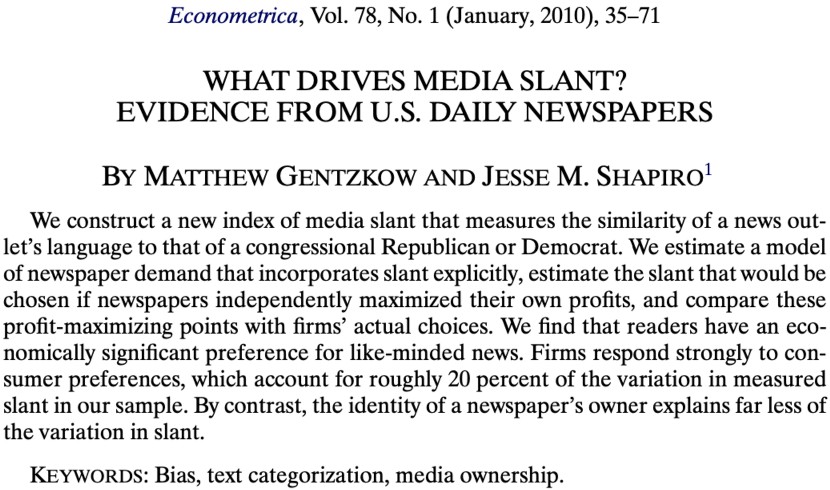
    ```
    nrounds (# Boosting Iterations)
    max_depth (Max Tree Depth)
    eta (Shrinkage)
    gamma (Minimum Loss Reduction)
    colsample_bytree (Subsample Ratio of Columns)
    min_child_weight (Minimum Sum of Instance Weight)
    subsample (Subsample Percentage)
    ```

  - Required packages: xgboost, plyr
  - A model-specific variable importance metric is available.

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.

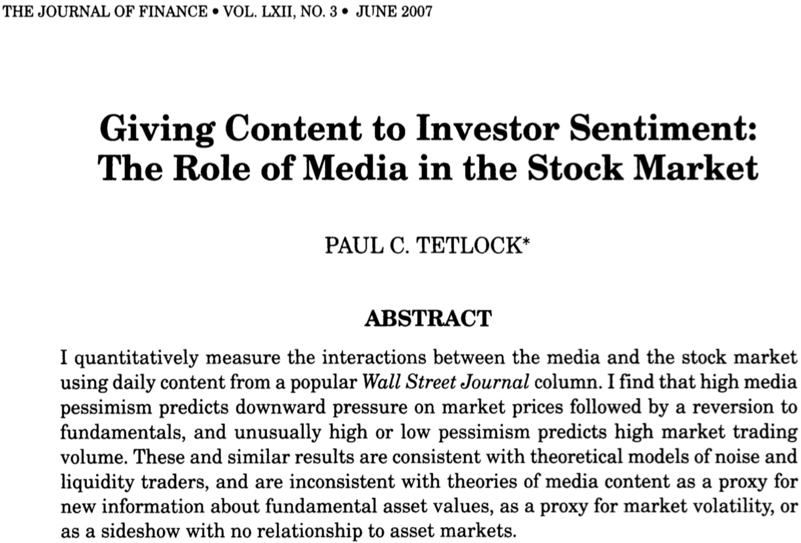
Text as Data: Motivation

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.

Information Retrieval and Tokenization

- 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.

Possible tokenization steps

- 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.
- Stemming: 'tax' ← taxing, taxes, taxation, taxable, ...
  A stemmer cuts words to their root with a mix of rules and estimation.'Porter' is standard for English.
- 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.
- Convert to lowercase, drop numbers, punctuation, etc ...
  Always application specific: e.g., don't drop `:-)` from tweets.

The *n*-gram language model

- An *n*-gram language model is one that describes a dialect through transition probabilities on *n* consecutive words.
- An *n*-gram tokenization counts length-*n* sequences of words.
  A unigram is a word, bigrams are transitions between words.
  e.g., `world.stage`, `stage.men`, `men.women`, `women.play`, ...
- This can give you rich language data, but be careful: *n*-gram token vocabularies are very high dimensional (p^n)
- More generally, you may have domain specific 'clauses' that you wish to tokenize.
- There is always a trade-off between complexity and generality.
- Often best to just count words.

Tokenization Demo

```r
## 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="~/Papers/Beauty_Hamermesh.pdf"), id=fname,
  language='en')
writeLines(content(notes)[1])
```

```
     ARTICLE IN PRESS
                                  Economics of Education Review 24 (2005) 369{376
                                                                                               www.elsevier.com/locate/econedurev
Beauty in the classroom: instructors’ pulchritude and putative
                                         pedagogical productivity
                                     Daniel S. Hamermesh, Amy Parker
                              Department of Economics, University of Texas, Austin, TX 78712-1173, USA
                                            Received 14 June 2004; accepted 21 July 2004
Abstract
   Adjusted for many other determinants, beauty affects earnings; but does it lead directly to the differences in
productivity that we believe generate earnings differences? We take a large sample of student instructional ratings for a
group of university teachers and acquire six independent measures of their beauty, and a number of other descriptors of
them and their classes. Instructors who are viewed as better looking receive higher instructional ratings, with the impact
of a move from the 10th to the 90th percentile of beauty being substantial. This impact exists within university
departments and even within particular courses, and is larger for male than for female instructors. Disentangling
whether this outcome represents productivity or discrimination is, as with the issue generally, probably impossible.
```

Tokenization Demo

```r
content(notes) <-iconv(content(notes), from="UTF-8", to="ASCII", sub="")

docs <- Corpus(VectorSource(notes))

names(docs) <- names(notes)

## 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
docs <- tm_map(docs, content_transformer(removeNumbers)) ## remove numbers
docs <- tm_map(docs, content_transformer(removePunctuation)) ## remove punctuation
## remove stopword.
##be careful with this: one's stopwords are anothers keywords.
# you could also do stemming; I don't bother here.
docs <- tm_map(docs, content_transformer(removeWords), stopwords("SMART"))

docs <- tm_map(docs, content_transformer(stripWhitespace)) ## remove excess white-space
```

Tokenization Demo

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

```
## <<DocumentTermMatrix (documents: 8, terms: 913)>>
## Non-/sparse entries: 1555/5749
## Sparsity           : 79%
## Maximal term length: 30
## Weighting          : term frequency (tf)
```

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

```
## <<DocumentTermMatrix (documents: 8, terms: 156)>>
## Non-/sparse entries: 650/598
## Sparsity           : 48%
## Maximal term length: 15
## Weighting          : term frequency (tf)
```

Tokenization Demo

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

```
## <<DocumentTermMatrix (documents: 5, terms: 8)>>
## Non-/sparse entries: 26/14
## Sparsity           : 35%
## ...
## Docs academic article beauty becker behavior biddle class classes
##    1        1       1      9      1        1      2     1       1
##    2        2       1      7      0        1      0     5       5
##    3        0       1      6      0        0      0     0       1
```

```r
## find words with greater than a min count
findFreqTerms(dtm,50)
```

```
## [1] "beauty"  "ratings"
```

```r
## or grab words whose count correlates with given words
findAssocs(dtm, "beauty", .7)
```

```
## $beauty
##  equation    effect     basic  positive     table perceived   results potential
##      0.86      0.83      0.79      0.77      0.77      0.77      0.73      0.72
##   problem   effects   instruc
##      0.72      0.71      0.70
```

Text Regression

- Once you have text in a numeric format, we can use all the tools we learned so far
- For example: Classify emails into spam

$$\text{logit}\left[\texttt{spam}\right] = \alpha + f\beta \tag{1}$$

- where $f_i = \frac{x_i}{\sum_j x_{ij}}$ are the normalized text counts

Text Regression: Example (Gentzkow and Shapiro)

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

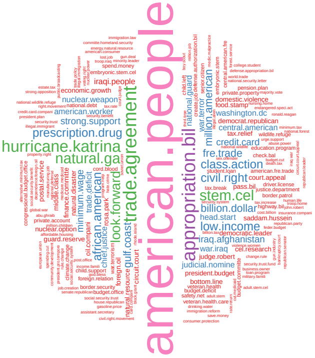
```
## 2 x 4 sparse Matrix of class "dgCMatrix"
##              stem.cel natural.ga hurricane.katrina trade.agreement
## Barack Obama        .          1                20               7
## John Boehner        .          .                14               .
```

```r
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: Example (Gentzkow and Shapiro)

```
require("wordcloud")
wordcloud(words = colnames(congress109Counts),
          freq = colSums(congress109Counts),
          min.freq = 100,
          scale = c(3, 0.1), max.words=200,
          random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Set1"))
```

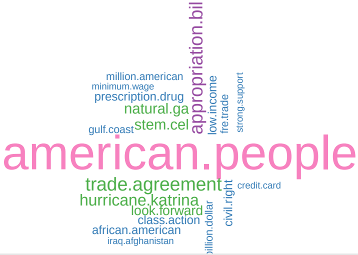


Text Regression: Wordle (Wordclouds)

```r
tail(colSums(congress109Counts))
```

```
##          stem.cel         natural.ga hurricane.katrina   trade.agreement
##              1699               1792              2020              2329
## appropriation.bil    american.people
##              2357               6256
```

```r
wordcloud(words = colnames(congress109Counts),
          freq = colSums(congress109Counts),
          min.freq = 1000,
          scale = c(3, 0.1), max.words=30,
          random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Set1"))
```



Text Regression

- We can use LASSO

```
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

```r
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
- there is often abundant *unlabeled* text
- 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

Topic Models: 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.

Topic Models: Example

- After cleaning and Porter 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
```

Topic Models: 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
```

Principal Component Analysis

- Dimensionality via main components

$$X = (x_1, x_2, \ldots, x_n)_{N\times K} \tag{2}$$

- Factor:

$$F = X\delta \;\; \delta \in K \tag{3}$$

- Idea: summarize the K variables in a single (F).
- Vocab: the coefficients of δ are the loadings: how much 'matters' each x s in the factor.
- Dimensionality: summarize the original K variables in a few $q < K$ factors.

Algebra Review

- Let $A_{m\times m}$. It exists
  - a scalar λ such that $Ax = \lambda x$ for a vector $x_{m\times 1}$,
  - if $x \neq 0$, then λ is an eigenvalue of A.
  - and a vector x is an eigenvector of A corresponding to the eigenvalue λ.
- $A_{m\times m}$ with eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_m$, then:

$$tr(A) = \sum_{i=1}^{m} \lambda_i \tag{4}$$

$$det(A) = \Pi_{i=1}^{m} \lambda_i \tag{5}$$

- If $A_{m\times m}$ has m different eigenvalues, then the associated eigenvectors are all linearly independent.
- Spectral decomposition: $A = P\Lambda P$, where $\Lambda = diag(\lambda_1, \ldots \lambda_n)$ and P is the matrix whose columns are the corresponding eigenvectors.

Factors via main components

- $x_1, x_2, \ldots, x_K$, K vectors of N observations each.
- Factor: $F = X\delta$
- What is the 'best' linear combination of $x_1, x_2, \ldots, x_K$?
- Best? Maximum variance. Why? The one that best reproduces variability original of all xs

Factors via main components

- Let
  - $X = (x_1, \ldots, x_K)_{N \times K}$,
  - $\Sigma V(X)$
  - $\delta \in K$
- $F = X\delta$ is a linear combination of X, with $V(X\delta) = \delta' \Sigma \delta$.
- Let's set up the problem as

$$\underset{\delta}{max} \;\; \delta' \Sigma \delta \tag{6}$$

  - It is obvious that the solution is to bring δ to infinity.

Factors via main components

- Let's "fix" the problem by normalizing δ

$$\begin{aligned} \underset{\delta}{max} \delta' \Sigma \delta \\ \text{subject to} \\ \delta' \delta = 1 \end{aligned} \tag{7}$$

- Let us call the solution to this problem δ^*.
- $F^* = X\delta^*$ is the 'best' linear combination of X.

Factors via main components

- Result: δ^* is the eigenvector corresponding to the largest eigenvalue of $\Sigma = V(X)$.
- $F^* = X\delta^*$ is the first principal component of X.
- Intuition: X has K columns and $Y = X\delta$ has only one. The factor built with the first principal component is the best way to represent the K variables of X using a single single variable.

Review & Next Steps

- XGBOOST
- XGBOOST demo
- Text as data
- Next class: More on text as data
- Questions? Questions about software?

Further Readings

- Gentzkow, M., & Shapiro, J. M. (2010). What drives media slant? Evidence from US daily newspapers. Econometrica, 78(1), 35-71.
- Friedman, J., Hastie, T., & Tibshirani, R. (2001). The elements of statistical learning (Vol. 1, No. 10). New York: Springer series in statistics.
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer.
- Taddy, M. (2019). Business data science: Combining machine learning and economics to optimize, automate, and accelerate business decisions. McGraw Hill Professional.