Lecture 25: Text as Data

Big Data and Machine Learning for Applied Economics
Econ 4676

Ignacio Sarmiento-Barbieri

Universidad de los Andes

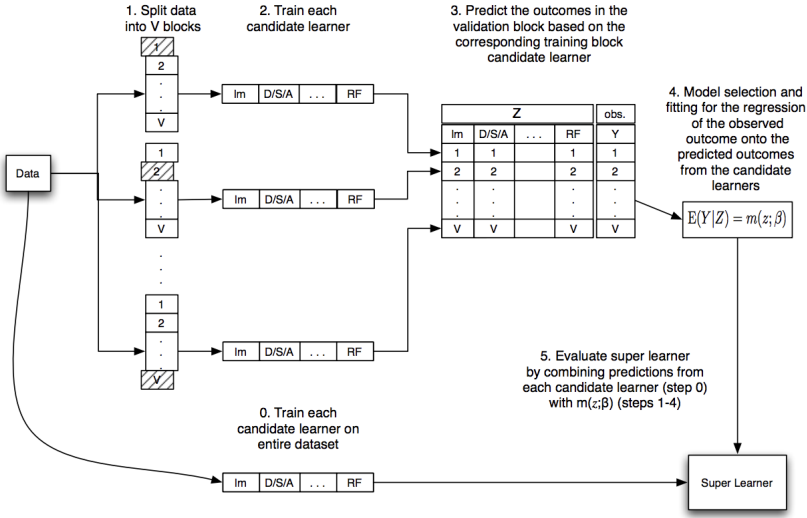
November 9, 2021

Agenda

Super Learnets

- Superlearning is a technique for prediction that involves combining many individual statistical algorithms to create a new, single prediction algorithm that is expected to perform at least as well as any of the individual algorithms.
- The superlearner algorithm "decides" how to combine, or weight, the individual algorithms based upon how well each one minimizes a specified loss function
- The motivation for this type of "ensembling" is that a mix of multiple algorithms may be more optimal for a given data set than any single algorithm.
- For example, a tree based model averaged with a linear model (e.g. random forests and LASSO) could smooth some of the model's edges to improve predictive performance.

Superlearnes: Summary



Source: Polley, Eric. Learning: Causal Inference for Observational and Experimental Data

Text as Data

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

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: 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

Text as Data: Motivation

- Jerry Moran, R-KS, says "death tax" relatively often and his district (Kansas 1st) voted 73% for George W. Bush in 2004.
- William Jefferson, D-LA, says "estate tax" relatively often and his district voted 24% for George W. Bush in 2004.

  $\Rightarrow$ "death tax" is republican

$$Ideology = f(\mathbf{X}_{\text{text}}) + u \tag{1}$$

where

$$ideology \approx g(Y_{Bush}) \tag{2}$$

- Gentzkow and Shapiro apply this logic to build an index of slant that sums across a speaker's term usage weighted by the direction of slant for each term.

Turn Text into Data: Tokenization

- Raw text—the language that humans read—is an incredibly rich object.
- It is much more than a sum of words.
- Meaning is built from ordered sequences of words that refer to each other, often separated across sentences or paragraphs.
- However, the vast majority of text analyses will ignore this complexity and rely solely on counts for language tokens: words, phrases, or other basic elements.
- Fortunately, a huge amount of information is contained in these simple counts.
- The lesson of the past 30 years has been that, for prediction and classification with text, it is difficult to make effective use of any statistics other than simple word counts.

Turn Text into Data: 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.

Turn Text into Data: Tokenization

- This is the Bag-of-Words representation of text.
- It treats documents as if they were constructed by randomly drawing words with-replacement from a bucket containing the fixed vocabulary.
- By summarizing a document in terms of word counts, we are throwing away all information relevant to any more complex processes of document construction.
- The advantage of this approach is its simplicity: you can characterize relationships between language and outside variables (e.g., authorship or sentiment) by building a model for word probabilities.

Turn Text into Data: Tokenization

Possible tokenization steps

- There are many possible algorithms for going from raw text to a summary token-count vector x.

  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...

- A possible sequence is:
  - 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.

- What the econometrian sees:

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

Turn Text into Data: Tokenization

Possible tokenization steps

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...

- Next
  - Stemming: 'play' ← play, players, ...
    A stemmer cuts words to their root with a mix of rules and estimation.'Porter' is standard for English.

- What the econometrian sees:

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

- Also, remove words that are super rare (in say $< \frac{1}{2}\%$, or $< 15\%$ of docs; this is application specific). For example, if Colombian occurs only once, it's useless for comparing documents.

The n-gram language model

- One common strategy is to count n-grams, or combinations of n words, rather than single "unigrams"
- 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.

Turn Text into Data: Tokenization

- Often best to just count words.
- For example, occurrences by party for some partisan terms

Congress	State	Party	America	Death Tax	Estate Tax	⋯
63	NM	dem	108	30	140	
		gop	100	220	12	

Turn Text into Data: DTM

- Once we have this clean text, we can convert from words to counts of words and create a document-term-matrix (DTM)
- X that has a row for each document and a column for each word.
- Element x_{ij} is the count for word j in document i.
- This is highly sparse (sometimes we remove terms that have zero counts more than 75% of the time)

Text Regression

- Once you have text in a numeric format, it will give us a powerful framework for text analysis.
- When we combine with all the tools we learned so far

$$y = f(word\ counts) + u \tag{3}$$

- where you can use ols, lasso, etc.

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 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

```
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
```

Factor 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

Factor Models

- One way to think about almost everything we do is as dimension reduction.
- We are trying to learn from high-dimensional x some low-dimensional summaries that contain the information necessary to make good decisions.
- Dimension reduction can be supervised or unsupervised.
- In supervised learning, an outside "response" variable y dictates the direction of dimension reduction.
  - In regression, a high-dimensional X is projected through coefficients β to create the low-dimensional (univariate) summary $\hat{y}$.

Factor Models

- In contrast, for unsupervised learning there is no response or outcome.
- We have a high-dimensional X, and you try to model it as having been generated from a small number of components.
- We are attempting to simplify X for its own sake.
- Why? For example, you might want to predict the sentiment of people from the words in their tweets; you will have a massive bank of all tweets (many X observations) but for only a small percentage will you know whether they are expressing positive or negative sentiment (e.g., by hand-labeling the tweets using human readers).
- An unsupervised analysis will use all of the tweets to break the content into topics, and then you can easily sort these topics by sentiment on the subset of labeled tweets.

Factor Models

- We'll explore different methods of factorization—tools that break the expectation for each X into the sum of a small number of factors.
- Let's then model the task

Factor Models

- We'll explore different methods of factorization—tools that break the expectation for each X into the sum of a small number of factors.
- Let's then model the task
- Given a matrix of high-dimensional data $X_{n \times p}$, you'd like to reduce this to a function of a few "important" factors.
- We do this by building a linear model for x as a function of these unknown factors and then estimating both factors and model at the same time.

Factor Models

- A factor model looks like

$$x_{ij} = \phi_{i1} f_{j1} + \cdots + \phi_{ik} f_{jk} \; j = 1, \ldots, p \tag{4}$$

- where
  - x_{ij} are the inputs of the regressions (independent vars)
  - The f_{ik} values are attached to each observation; they are like the x_{ij} inputs in regression, except that they are now unknown latent factors that need to be estimated.
  - ϕ_{jk} coefficients are called loadings or rotations—these are properties of the model and are shared across all observations. They are coefficients for regression of x_{ij} onto f_i.
  - When you use a K that is much smaller than p, factor models provide a parsimonious representation for X.
  - Each observation x_i is mapped to K factors $\phi_{i1} \ldots \phi_{ik}$, and these factors are a low dimensional summary of X.

Factor Models

- How do you estimate a Factor Model?
- You need to regress X onto f,
- which would be easy except that the f are latent: you don't know them and they need to be estimated.
- It turns out that there are a number of fast ways to estimate the model
- One procedure is called principal component analysis (PCA).
- The result of PCA is a set of rotations $\phi = (\phi_1, \ldots, \phi_k)$. These can be used to obtain the factor scores f_i for any observed x_i.

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
```

Review & Next Steps

- Text as Data: Tokenization
- Text Regression w. Example
- Factor Models: Example PCA
- Next class: More on text as data and PCA
- 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.