Text as Data
Vector Space Models

…and Sentiment Analysis

Robert Stine
Department of Statistics
Wharton School of the University of Pennsylvania

www-stat.wharton.upenn.edu/~stine

Comments from First Lecture

Preparing text

Depends on nature of the analysis

- For example, to remove or keep stop words or capitalization

Bag-of-words representation

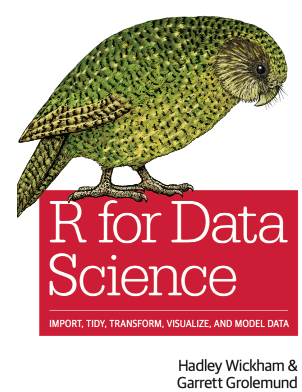
- Document-term matrix sacrifices the order of text

NLP: deeper linguistic analysis

- Identify named entity, parts of speech, grammatical structure
- Language specific, unlike DTM approach with symbol counts

Tidy R

- It's different, so check out R for Data Science by Wickham and Grolemund



Slides and Rmd file

Edits often happen after the lecture!

- Files stay on website so no need to grab right away

Sentiment Analysis

Sentiment Analysis

Typical approach

Start with dictionary of words associated with concepts

Positive - Negative
Cruel - Kind
Red - White wine

Over a corpus of documents, count the prevalence of the different types of words

Use prevalence of these counts to measure of the “sentiment” of the document

Application

Words used by judge hearing a case, speeches, social media

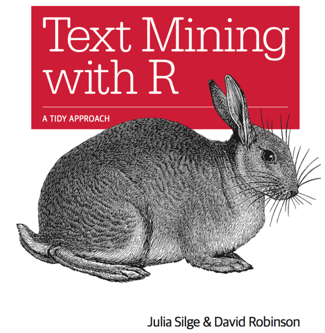
Dictionaries

Dictionary also called a lexicon

Four examples

Included in the R package tidytext

Text Mining with R, a Tidy Approach (2017) Silge and Robinson



Bing

The classic: positive and negative words, binary categorical coded

NRC

More “emotions” beyond just positive or negative

Anger, anticipation, disgust, fear, joy, sadness, surprise, and trust

AFINN

Numerical scores for positive/negative from -5 to +5; others are categorical

Loughran

Special purpose for financial terms

Examples

Bing

2069	envy	positive
2070	epidemic	negative
2071	equitable	positive
2072	equivocal	negative
2073	erase	negative
2074	ergonomical	positive
2075	erode	negative
2076	erodes	negative

AFINN

1586	n00b	-2
1587	naive	-2
1588	nasty	-3
1589	natural	1
1590	needy	-2
1591	negative	-2
1592	negativity	-2
1593	neglect	-2

NRC

9677	prestige	positive
9678	prestige	trust
9679	presto	joy
9680	presto	positive
9681	presto	surprise
9682	presumption	trust
9683	presumptuous	anger
9684	presumptuous	disgust
9685	presumptuous	negative

Loughran

2129	ingenuity	positive
2130	inhibit	constraining
2131	inhibited	negative
2132	inhibited	constraining
2133	inhibiting	constraining
2134	inhibits	constraining
2135	inimical	negative
2136	injunction	negative
2137	injunction	litigious

Reactions to these dictionaries?

Formation of Dictionary

Generic

One size fits all: dictionary may become “dated” or unsuited to your data, such as language used in social media, emoticons

Dictionaries tend to be dominated by negative words

Bag of words

Counts “beautiful” same as “not beautiful”.

Sarcasm is hard to measure.

Grow your own

Expand using WordNet to find synonyms, antonyms

Supervised data needed, but hard to come by

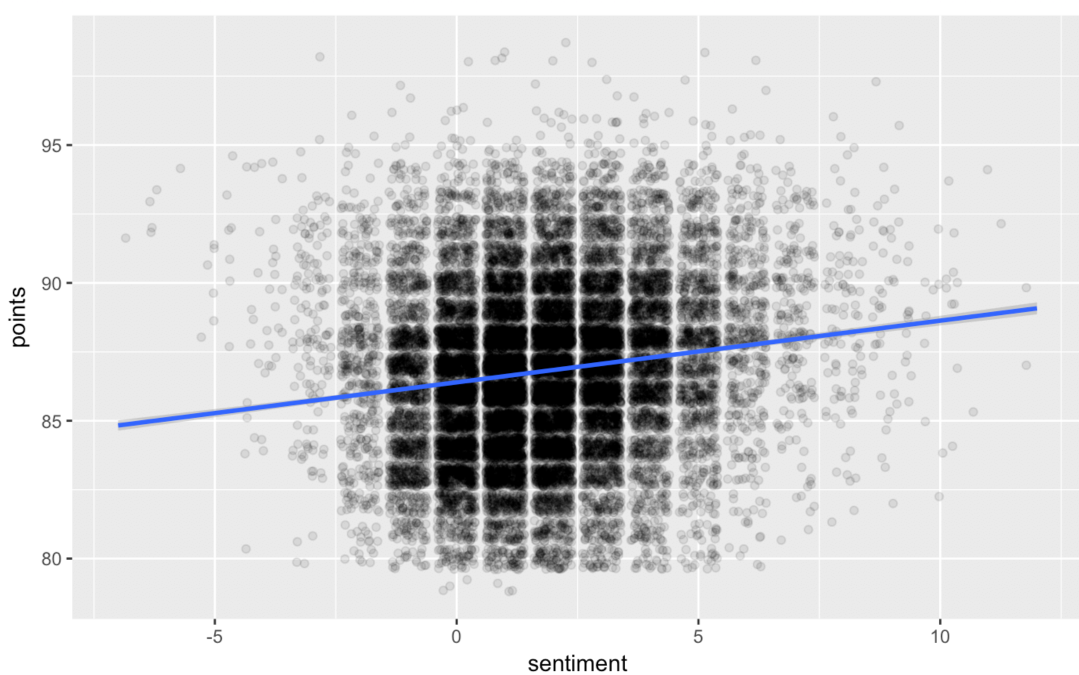
Example with Wines

Relate counts of words to points assigned to wines

Is "lemon" a negative word when describing wine?

Use counts or proportions

Net sentiment weakly related to points



est points ≈ 86 + 0.2 sentiment

RMSE ≈ 3

$R^2 \approx 2\%$

What's a big assumption?

Weaker than using similar word lists.

Combination

Multiple regression

Allows different effects for positive and negative words

Include nonlinear terms add a bit more

Requires a response to judge the effects of sentiment words

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	84.800892	0.063968	1325.670	< 2e-16	***
rating.pos	0.582570	0.032998	17.655	< 2e-16	***
rating.neg	0.293899	0.054162	5.426	5.82e-08	***
I(rating.pos^2)	-0.014903	0.004193	-3.554	0.00038	***
I(rating.neg^2)	0.096777	0.011990	8.072	7.31e-16	***
rating.pos:rating.neg	0.019317	0.011319	1.707	0.08791	.

```
Residual standard error: 2.883 on 20323 degrees of freedom
  (179 observations deleted due to missingness)
Multiple R-squared:  0.137,     Adjusted R-squared:  0.1368
```

Discussion

Sentiment analysis requires a dictionary

- Assigns a fixed set of weights to words
- Unsupervised
  - Not what you would find from a dummy variable regression, but regression would require you to have a response variable
  - The R notes contain an very quick look at how you can use a response (the rating points in this case) to set weights.
- Dictionaries are dated and often context dependent
  - “lemon” is not a bad word in one’s sentiment toward wine

Experiment with other dictionaries

- Only shown results from the oldest, simplest dictionary
- Accompanying R shows“how its done”

Latent Semantic Analysis

Recall

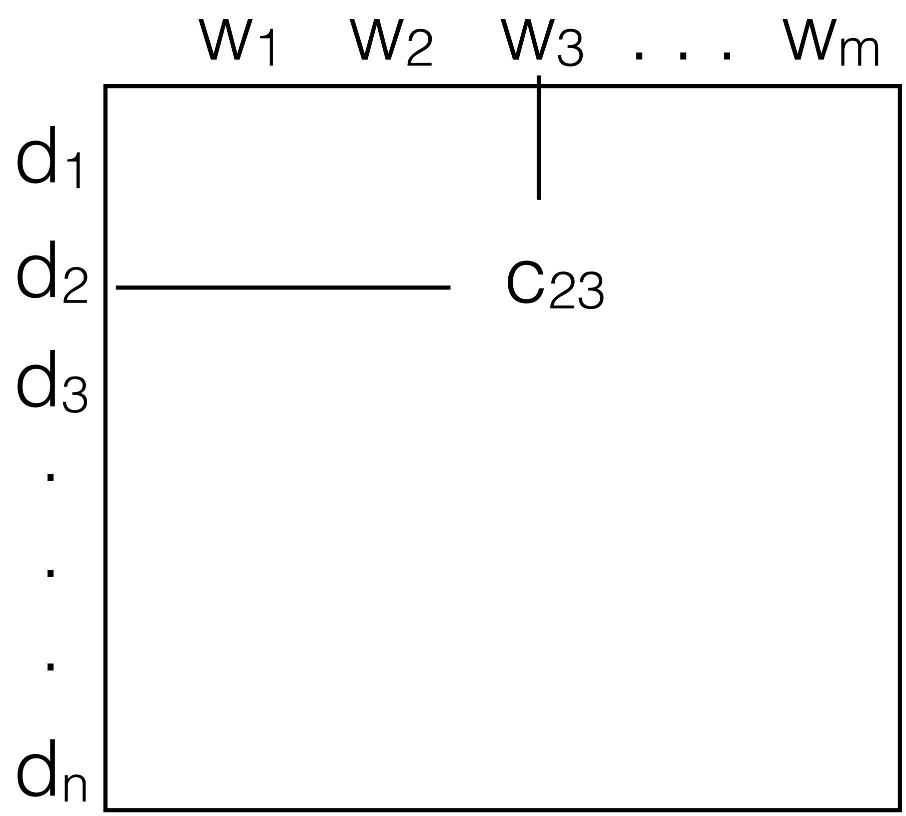
Document Term Matrix

Count word types that appear in each document

One row for every document (an observation)

One column for every word type (a variable)

C =

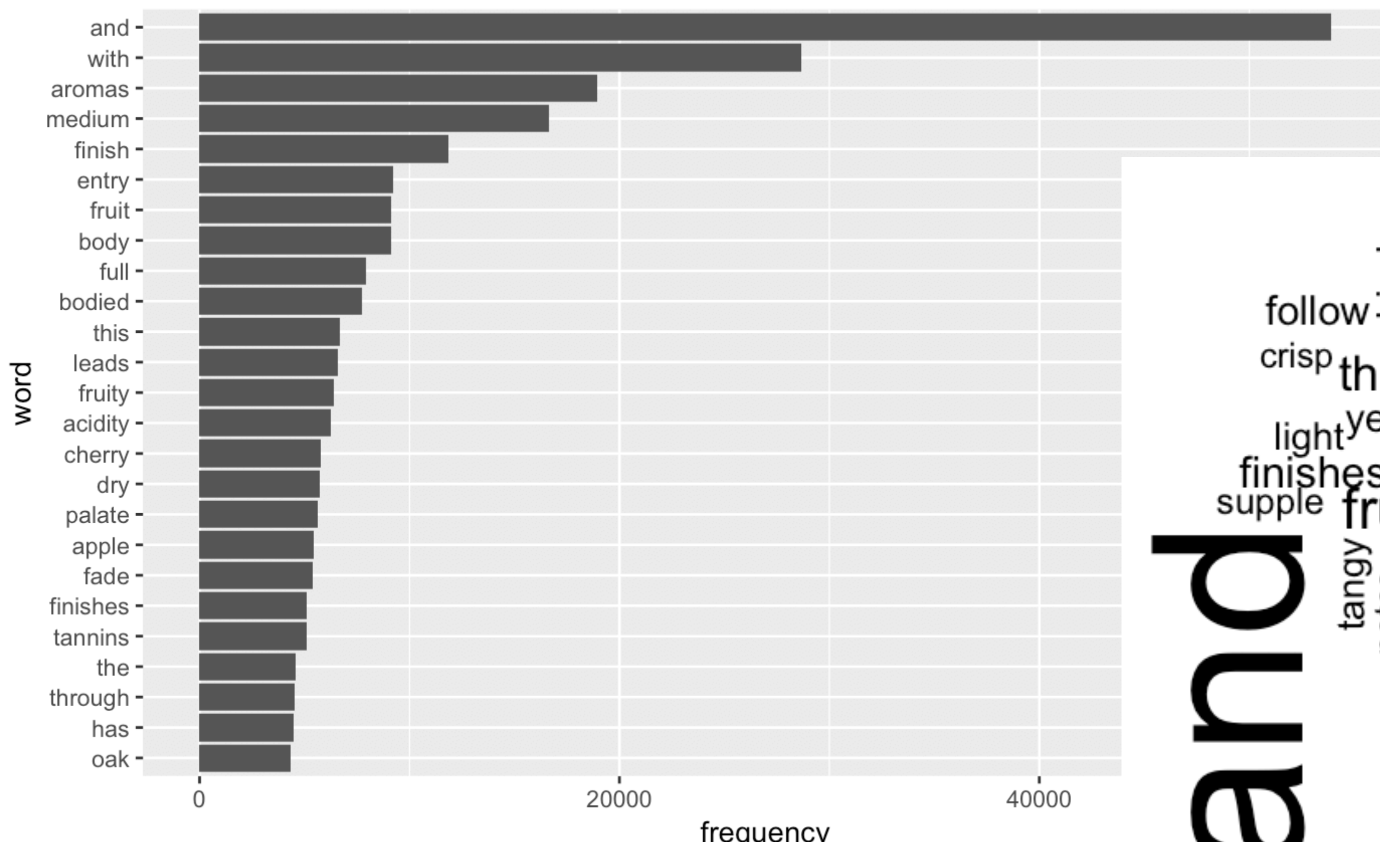


number of times word type w_3 appears in document 2

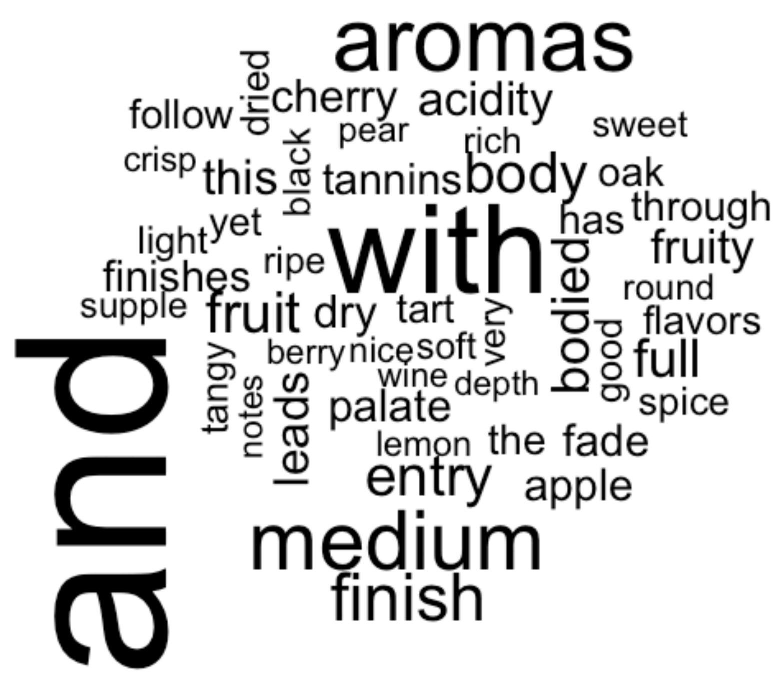
Popular Summary Plots

Bar charts and word clouds are popular graphs used to summarize frequencies of word types

Column totals from the document-term matrix



Nicer without those stop words…



Distribution of Types

Most word types are rare, most tokens are common

Total of 607,355 tokens from 5,488 word types

Zipf distribution for word types

Depends on how text was tokenized

Power law has ideal form…

Frequency of second most common 1/2 frequency of most common

Frequency of third most common 1/3 frequency of most common…

$f_j = (1/j)\ f_1,\ j = 2,3,4\ldots$

Highly skewed (plot follows)

Most common types include stop words and words related to wine: aromas, body, dry, palate, acidity, fruit, tannins.

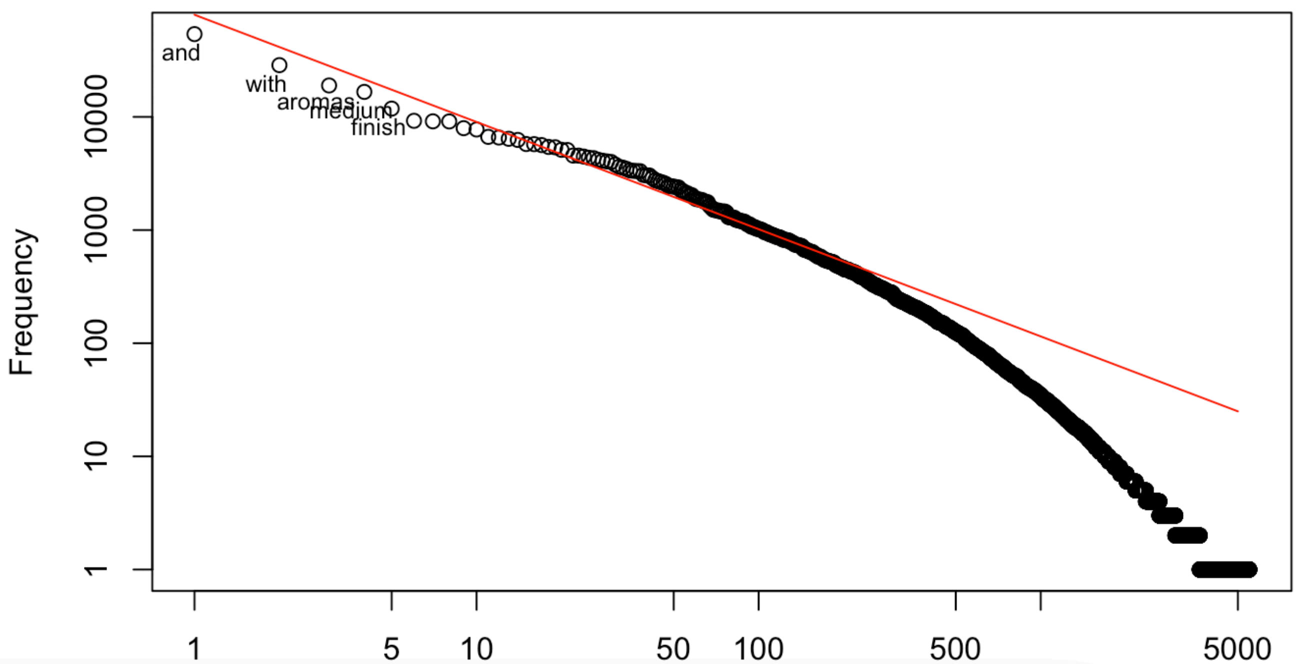
Distribution of Types

Plot log of frequency on log of rank

Sum columns of C, ordered by frequency

Power law would be a line

Most data produce this concave shape



slope for first 250 is -0.95

Discussion of DTM

Sensitive to subjective choices of analyst

How was the text tokenized?

Bag-of-words

bag: A collection of elements that allows copies

A set is a special case of a bag that limits each count to 1.

Each row of C (one document) is a bag.

Sequence order is lost: Random permutations of the tokens produce the same document-term matrix.

Sparse representation is essential

C is 20,508 x 5,488, with about 112 million elements

Common vocabulary might have 50,000 word types

Handling Rare Types

What to do about rare word types?

- 1827/5488 ≈ 33% of word types appear just once!
- Another 660 + 367 = 1027 appear just 2 or 3 times

Anticipate complication

- Suppose we use word counts to predict price of wine
- Split sample analysis: say, half for modeling, half for testing
- Test sample guaranteed to have words we never saw in building our model and possibly omit words in model

Recode as out-of-vocabulary (OOV)

- Just one symbol, or distinguish depending on use in context?

Handling Rare Types

Possible ways to reduce number of OOVs

Stem the words: “cigars” found 1 time, “cigar” found 152

But does “fruit” == “fruity”?

Fix spelling errors: “berrry", “ciitrus”

Combine numbers as one type of OOV

Recoding as OOV

Can use a special OOV for numbers

Part of speech tagging
Special OOV for nouns vs verbs vs places vs things etc

Losing sight of forest for trees?

603,107 tokens represent types seen more than 3 times

4,248 seen 3 or less

Latent Semantic Analysis

Principal components analysis of the document-term matrix (or possibly a bigram matrix)

- Actually closer to canonical correlation analysis
- Heuristic: Words that appear together are related, the so-called distributional hypothesis

Applications: supervised or unsupervised

- Supervised: Build features for predictive models
- Unsupervised: embedding
  - LSA represents document as point in R^d, dimension d << m
  - Preserves distances between documents, but in lower dim
  - Coordinates taken from PCA of standardized DTM

Process Overview

Start from a matrix of counts

- Document term matrix: count types that occur in same document
- Bigram matrix: count types that appear adjacent to each other

Compute principal components from matrix

- Requires standardization
- DTM, bigram matrices interpretable as covariance matrices

Principal components define "word embedding"

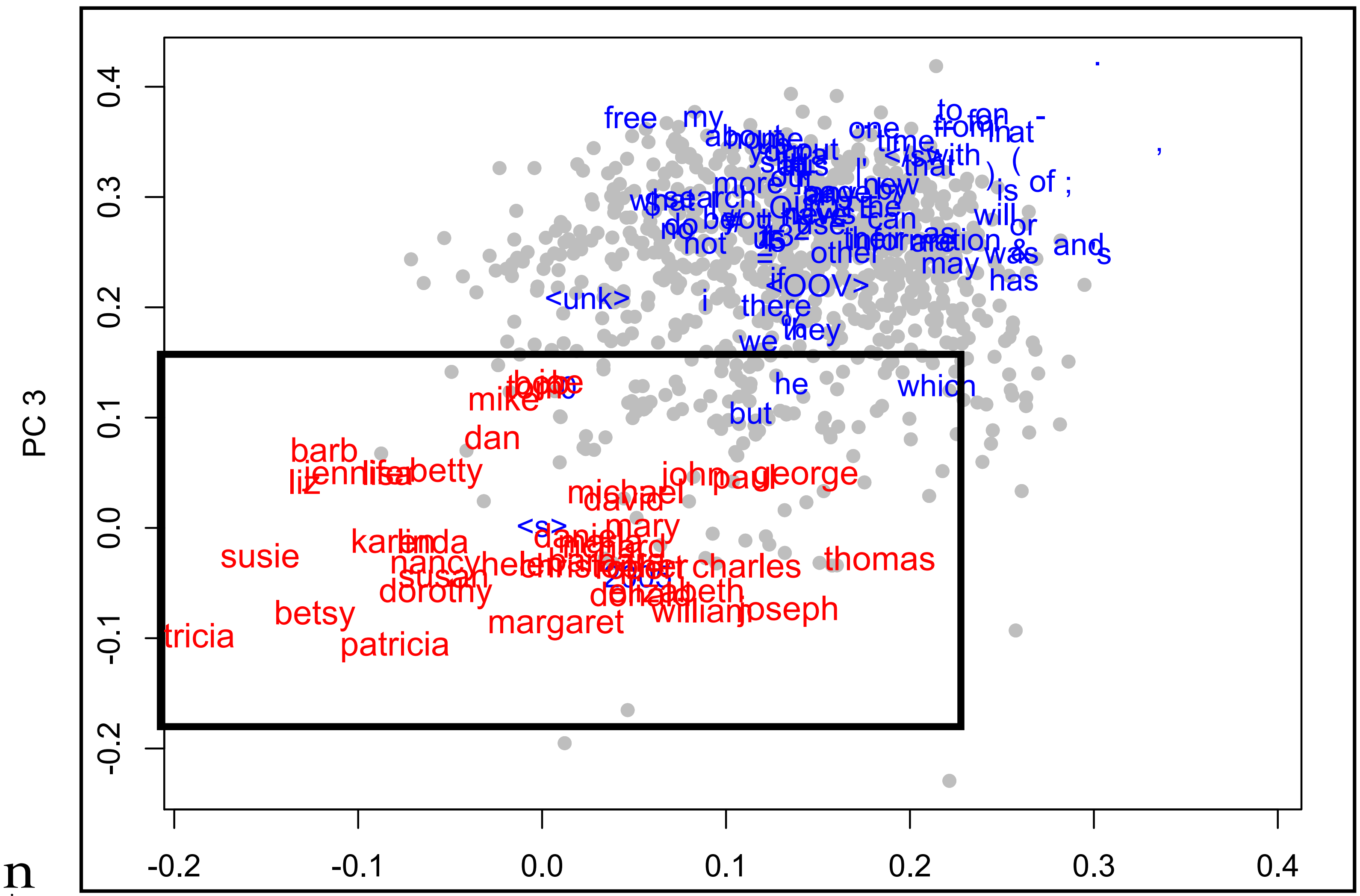
- Coordinates of similar words appear near each other

Variables may then be used in other models

Examples of Embeddings

Plot two dimensions from the word “embedding”

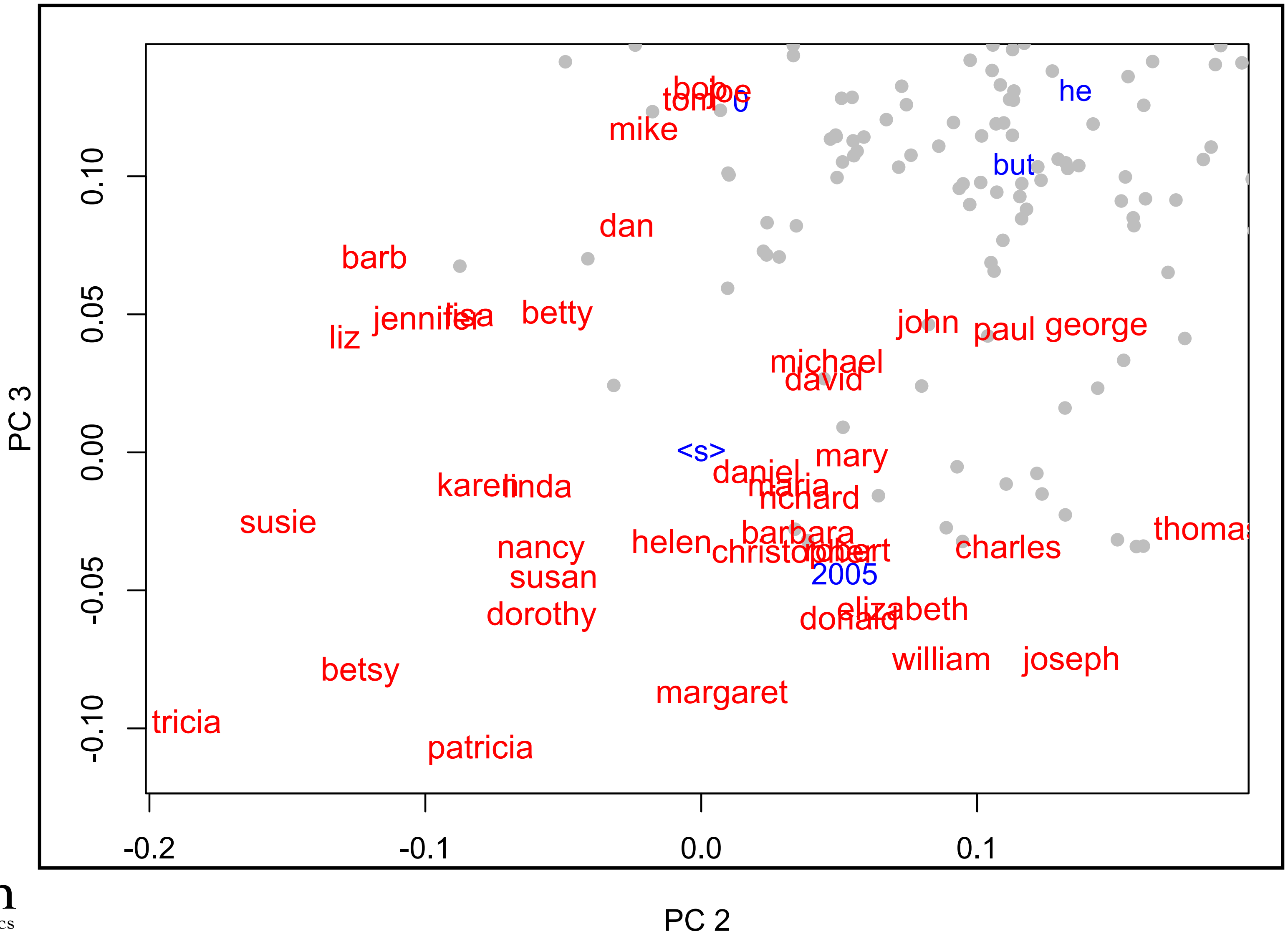
Based on data from Google bigrams



Wharton
Department of Statistics

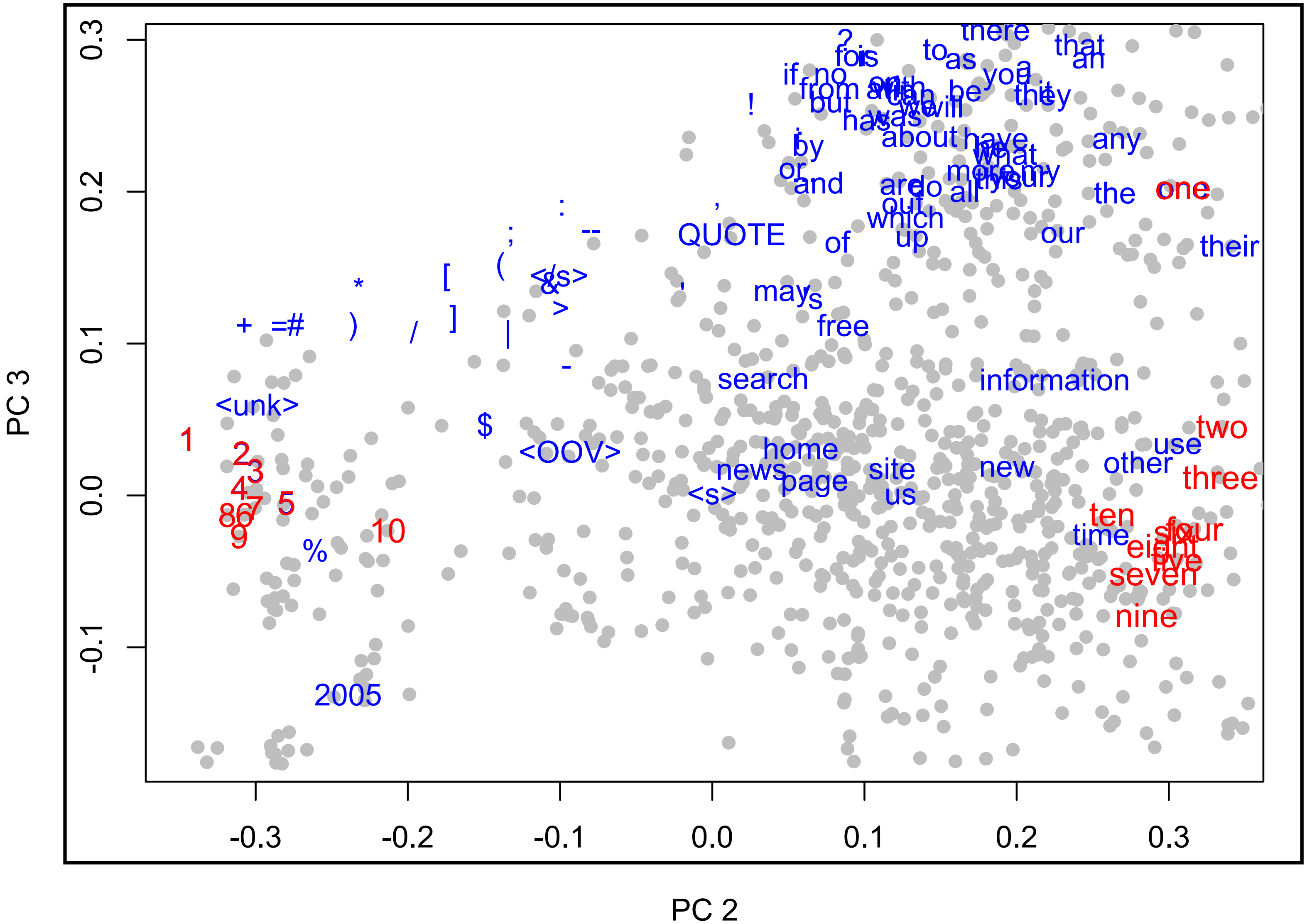
Examples of Embeddings

Zoomed in view of same singular vectors



Examples of Embedding

Numbers as words and digits



Closer Look at LSA

LSA ≈ PCA of document-term matrix C (or bigram)

Conceptual motivation

- Distributional hypothesis: Word types that are used in the same way (same context) have similar meaning
- Each document is a mixture of themes or "topics" that dictate word usage (see explicit model tomorrow)

Concerns

- How to standardize the variables
  - PCA is most sensible when variables have been standardized.
  - Not sensible to make columns of C have equal SD (remember sparsity)?
- PCA designed for a multivariate normal world. C is sparse

Conventions for LSA

Centering columns of C

Not done. Counts are all positive with mean near zero.

Scaling columns of C is interesting

Length normalization

Reduce the influence of longer documents, replacing

$C_{ij} \text{ –> } C_{ij}/n_i$ or possibly $C_{ij} \text{ –> } C_{ij}/\text{sqrt}(n_i)$

Term frequency - inverse document frequency (tf-idf)

Give more weight to words that are common in a document (tf), but not so common elsewhere (idf).

Let d_j denote the number of documents in which w_j appears.

$C_{ij} \text{ –> } C_{ij} \text{ x } \{\text{\# docs}\}/\{\text{\# } m_j \neq 0\}$

Combinations, such as

$C_{ij} \text{ –> } \log(1+ C_{ij}) \text{ x } \log(\{\text{\# docs}\}/\{\text{\# } m_j \neq 0\})$

Token Space

Novel perspective on the document-term matrix

Consider two matrices with elements 0 and 1

Total number of rows = total number of word tokens

Stochastic process

N = total # tokens
n = # documents
m = # word types

documents

$$
S = \begin{array}{c|ccccc|}
 & d_1 & d_2 & d_3 & \dots & d_n \\
\hline
t_1 & 1 & 0 & 0 & & 0 \\
t_2 & 1 & 0 & 0 & & 0 \\
t_3 & 1 & 0 & 0 & & 0 \\
\hline
t_4 & 0 & 1 & 0 & & 0 \\
t_5 & 0 & 1 & 0 & & 0 \\
\vdots & & & & & \\
 & 0 & 0 & & & 1 \\
t_N & 0 & 0 & & & 1 \\
\hline
\end{array}
$$

word types

$$
W = \begin{array}{|ccccc|}
w_1 & w_2 & w_3 & \dots & w_m \\
\hline
0 & 0 & 1 & & 0 \\
1 & 0 & 0 & & 0 \\
0 & 1 & 0 & & 0 \\
0 & 1 & 0 & & 0 \\
0 & 0 & 0 & & 1 \\
 & & & & \\
0 & 0 & 0 & & 0 \\
\hline
\end{array}
$$

preserves **all** information in source text

Wharton
Department of Statistics

DTM ≈ Covariance

Document-type matrix is nxm matrix

$$S^T W = C$$

Counts of the word types in each document

$$C_{ij} = \#\{w_j \text{ in } d_i\}$$

View columns of S and W as indicator variables

Because most types are rare, means ≈ 0 and

$$C_{ij} \approx N \operatorname{cov}(d_i, w_j)$$

Standardize binomial variation

Document counts: $\operatorname{var}(D_i) = (n_i/N)\,(1-n_i/N) \approx n_i/N$

$C_{ij} \to C_{ij}/\operatorname{sqrt}(n_i m_j)$

Word type counts: $\operatorname{var}(W_j) = (m_j/N)\,(1-m_j/N) \approx m_j/N$

Canonical Correlation Analysis

Extension of regression to multivariate Y

Regression
Find the linear combination of the columns of X that is most correlated with Y

CCA
Find the linear combination of the columns of X that is most correlated with a linear combination of the columns of Y

Role in text

Binary matrices S and W play roles of Y and X

Complication: computation

CCA requires standardization of X and Y

Implies inversion of m x m and n x n matrices (e.g., $(X^TX)^{-1}$)

Singular Value Decomposition

Decompose any matrix into orthogonal pieces

Assume X is an n x m matrix of rank d ≤ min(n,m)

$$\underset{n \times m}{X} = \underset{n \times d}{U}\,\mathrm{diag}(d_j)\,\underset{d \times m}{V^T} = \Sigma\, d_j\, u_j\, v_j^T$$

where U and V are orthogonal

$$U^TU = I_d,\ V^TV = I_d$$

U = “components”
V = “loadings”

Rank(X) = Number singular values $d_j \neq 0$

spectrum

Collection of singular values known as “spectrum” of X

Caution: Outliers will be important

SVD is a squared-error approximation

Wharton
Department of Statistics

Interpreting the Components

General approach

Plot components versus each other: often see clusters

Plot components versus other known variables

Plot loadings with labels of important word types

Rotation

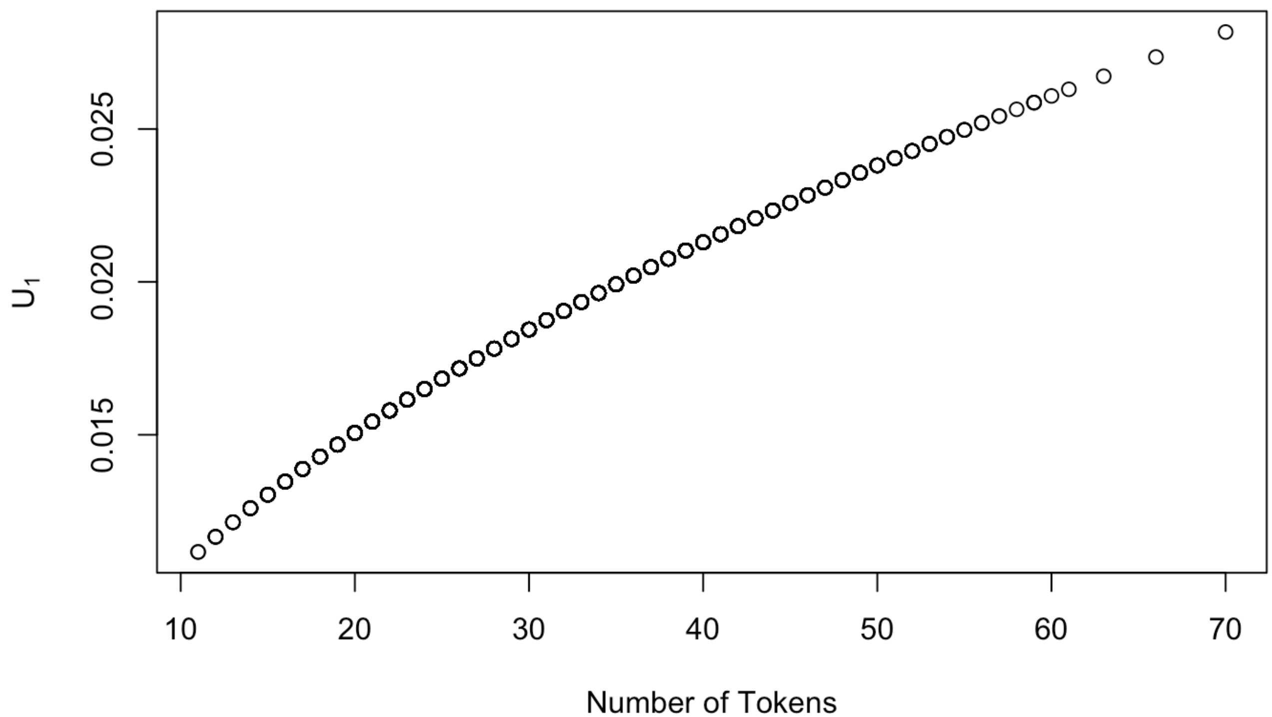
Can be used as in principal components to obtain a simpler structure to the coefficients (e.g., Varimax rotation)

Less commonly see in text, though found in JMP

Example from Wines

First component

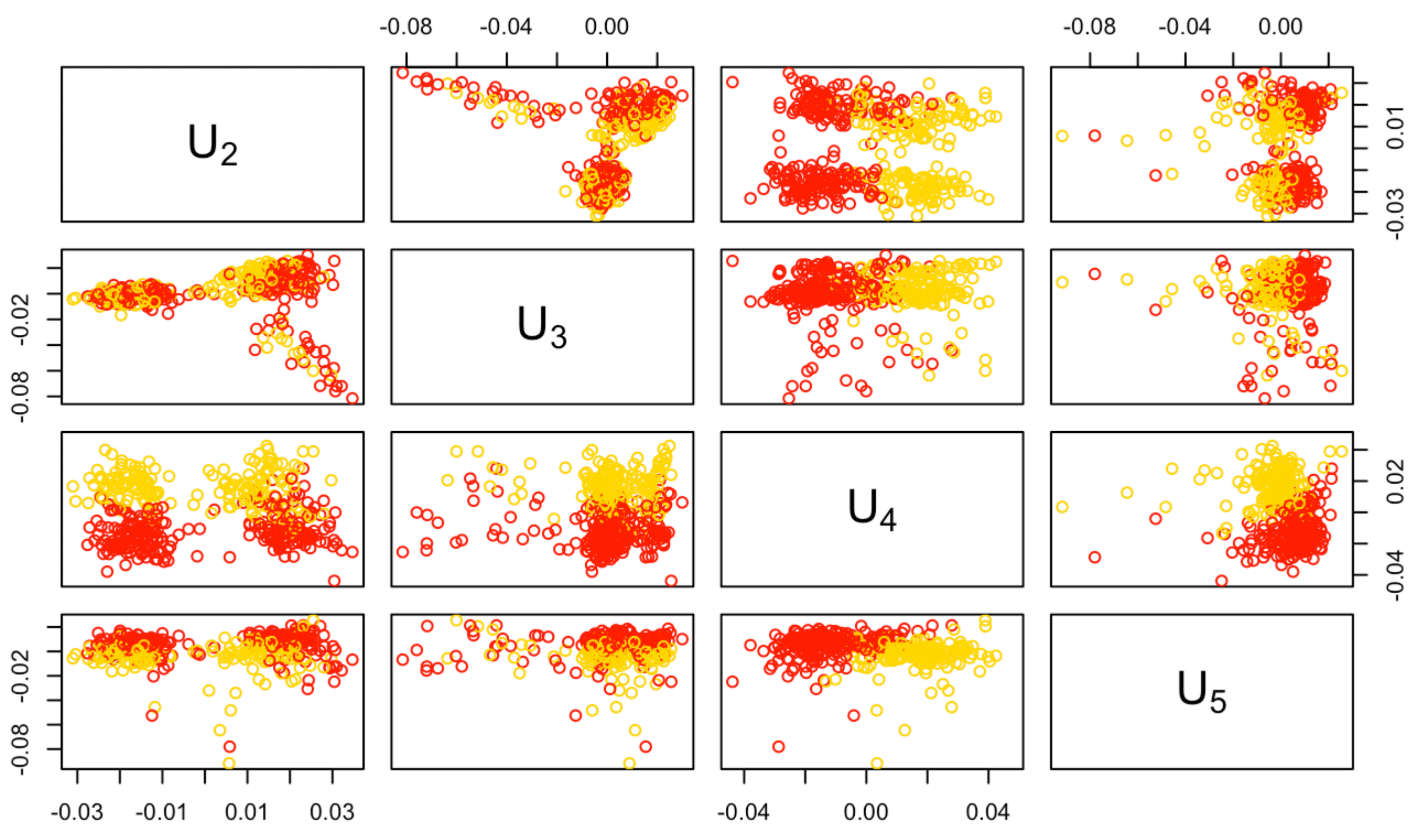
The first component when using CCA normalization of the wines measure the number of tokens in the document.



Example from Wines

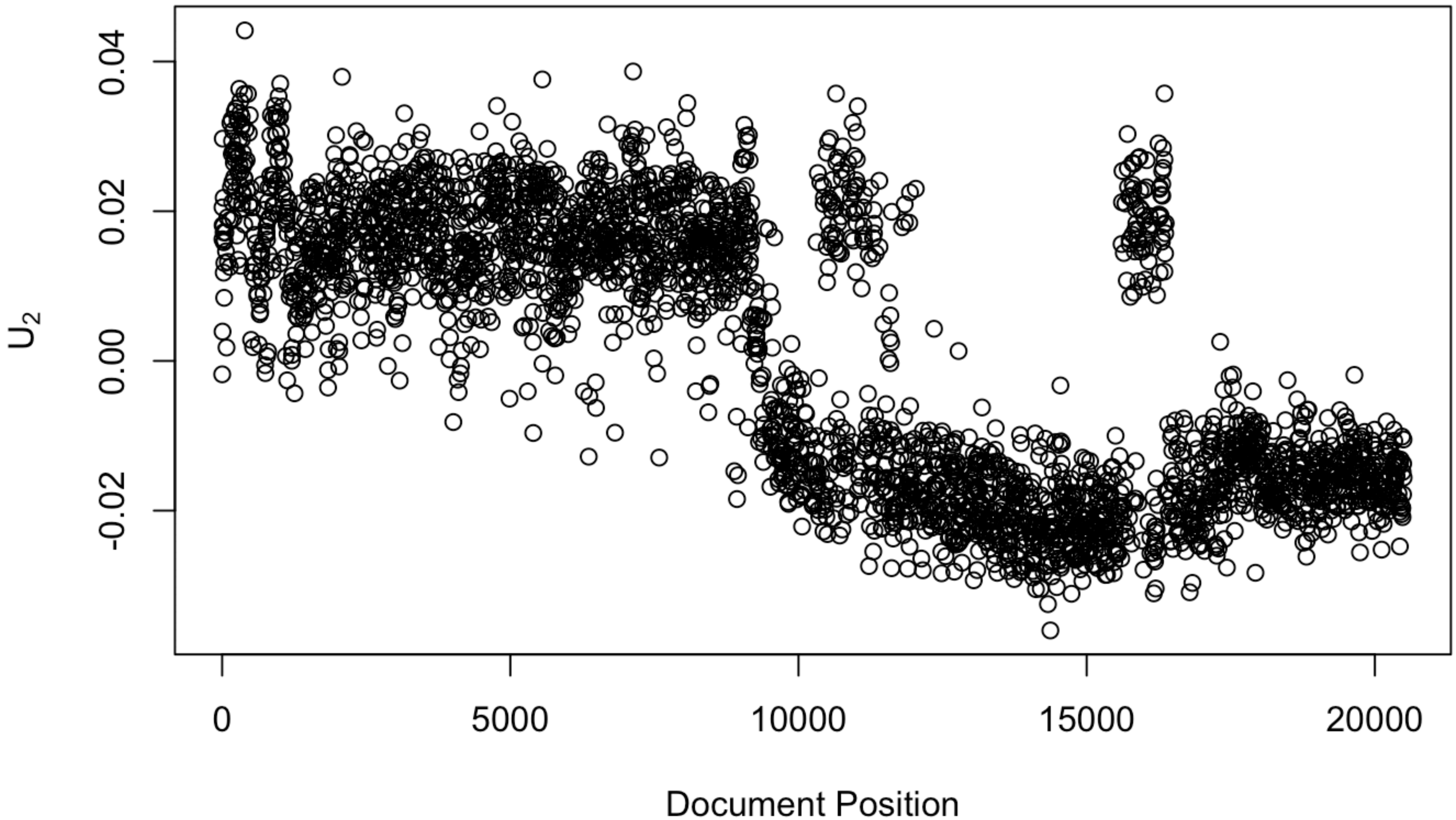
Principal components reveal clusters unrelated to wine color or variety…

Just the same, easy to use U_4 to predict the wine color.



Example from Wines

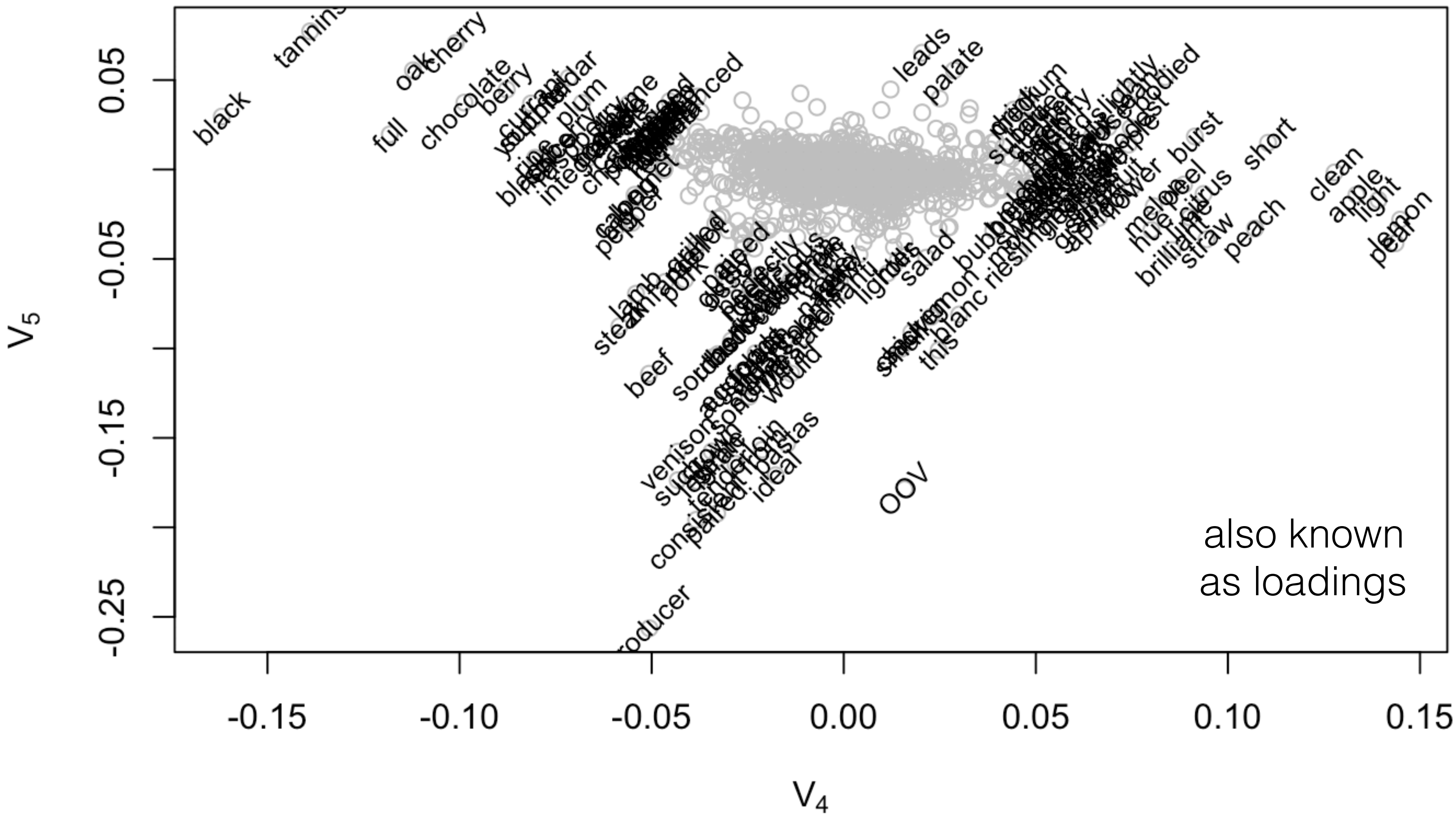
But a sequence plot shows a clear pattern…



We will see what happened in the R session.

What are those components?

Key words that comprise two components that separate the wine colors.



Random Projection

Recent development

Reduce the number of columns of a matrix by multiplication by a random matrix (yes, a matrix of random numbers)

Preserves much of the “structure” of the matrix, in particular, the column span, distance matrix, and bigger principal components

SVD by random projection

Reduces the number of columns from thousands to 100s

Reproduces the SVD in examples when you can do the calculations in R

Algorithm

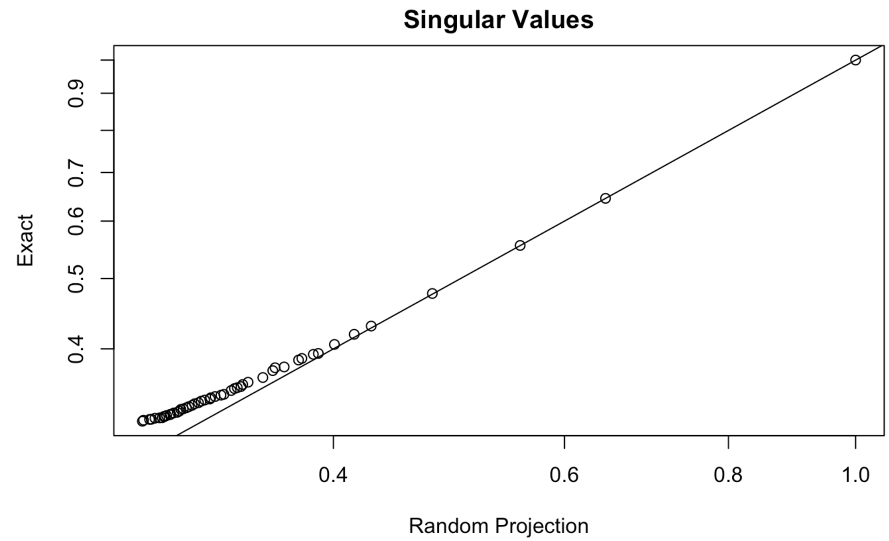
Power iterations improve recovery

Demostration with Wines

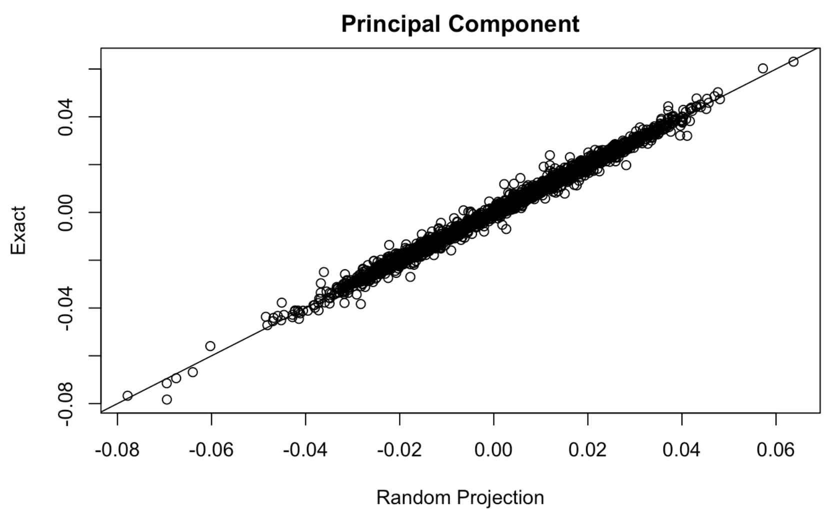
Random projection captures spectrum

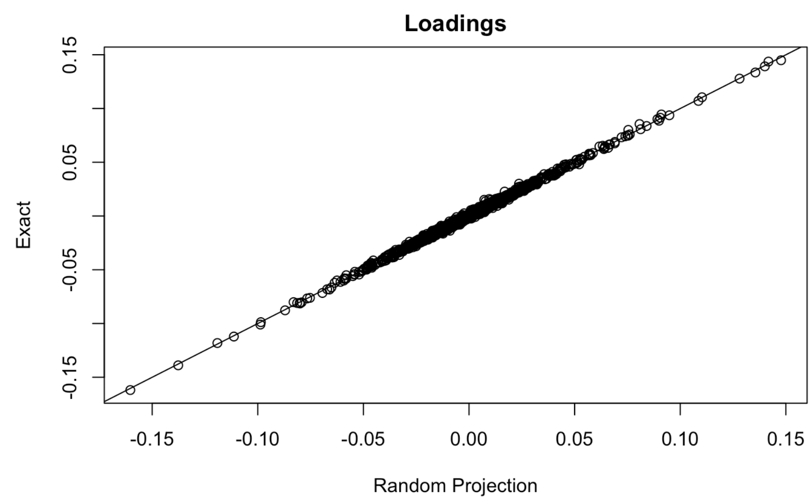
Compare singular values and and coefficients U and V

Use “small” problem in which R can do the exact decomposition



And coordinates of components





Discussion

Learning more

LSA is just a button click away, but there's much to learn about what's happening under the hood.

Don't need to be an expert mechanic to drive a car, but helps to have an idea of what's going on.

Some papers

Deerwester, et al (1990). Indexing by latent semantic analysis. JAsIs, 41, 391-407

Landauer, Foltz, and Laham (1998). An introduction to latent semantic analysis. Discourse Processes, 25, 259-284

Schwarz, Turney and Pantel (2010). From frequency to meaning: vector space models of semantics. J. of Artificial Intelligence Research, 37, 141-188