# Vector Space Models

wine_spectral.R

# Latent Semantic Analysis

## Problem with words

Even a small vocabulary as in wine example is challenging

## LSA

Reduce number of columns of DTM by principal components

Enables algorithms that are otherwise impractical (eg, cluster)

Offers potential interpretations

## Embedding

DTM represents a document as a sparse vector of m counts

LSA represents a document as a point of dimension d << m

Preserves distances between documents, but in lower dim

# Principal Components Analysis

Find weighted linear combinations of variables that

(a) have maximal variation

(b) are uncorrelated with each other

## Typical role

Reduce a large collection of features to a smaller number of uncorrelated variables, the principal components

Preserve most of the variation and correlations

## Computation

weighted sums of variables only make sense when standardized to common scale

X = matrix of centered and standardized data (mean 0, sd 1)

Sample covariance/correlation matrix of X is $S_{XX}$

Leading eigenvector of $S_{XX}$ is first principal component

# PCA Algebra

## First principal component

Maximizes the variance of a linear combination of the columns of the variables $X = X_1, X_2, \ldots, X_k$

$$\max_z \text{var}(X^T z) = z^T S_{XX} z$$

Solution is first eigenvector of the covariance matrix of X

$$z = e_1, \quad \text{where} \quad S_{XX} e_1 = \lambda_1 e_1$$

## Second principal component

Second eigenvector, $S_{XX} e_2 = \lambda_2 e_2$ , $\lambda_2 < \lambda_1$

Orthogonal to first, $e_1^T S_{XX} e_2 = 0$

## Full decomposition

$$\underset{n \times k}{E} = (e_1, e_2, \ldots, e_k) \qquad S_{XX} E = E \, \text{diag}(e_j)$$

# Terminology

## Loadings

Coefficients that define the weighted sums

Eigenvectors of the covariance matrix of X

Describe how much of each component of X goes into weighted sums

Xs need to have a common scale

## Scores

The weighted combinations defined by the loadings

Uncorrelated

The variables used in principal components regression

# Dimension Reduction

## Ideal scenario for PCA

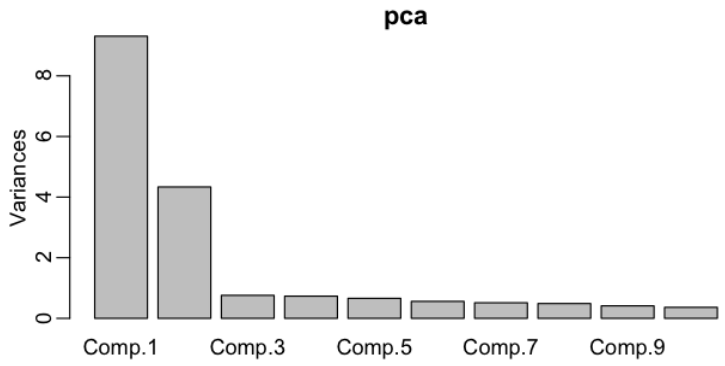
## Latent variable

Observe noisy versions of underlying latent variables

$$X_j = L_j + \text{random noise}$$

If only one latent variable, could find it by simply averaging the X's, but that seems too easy.

## Spectrum of matrix (scree plot)



# Problem for PCA

Suppose X has a very large number of columns (m>n)

- A document-term matrix has thousands of columns

Calculation of correlation matrix is slow

- Size of covariance matrix increases as square of number of variables
- Number of columns will exceed number of documents in some cases

Modern approaches

- Avoid the calculation of covariance matrix
- Singular value decomposition
- Random projection

# Singular Value Decomposition

Decompose any matrix into orthogonal pieces

Avoids explicitly computing the covariance matrix

Assume X is an n x m matrix of rank $d \leq \min(n,m)$

$$X = U\,\mathrm{diag}(d_j)\,V^T = \Sigma\, d_j\, u_j\, v_j^T$$

n x m  n x d  d x m

where U and V are orthogonal

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

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

# PCA via the SVD

## Recall characterization of PCA

Need to find matrix E s.t.

$$E^T S_{XX} E = \text{diag}(\lambda) \quad \text{or} \quad S_{XX} = E \, \text{diag}(\lambda) \, E^T$$

## Start with SVD of X

Assume X has been centered to have mean zero

$X = U D V^T$

$n S_{XX} = X^T X = (UDV^T)^T UDV^T = V D^2 V^T$

Elements of V are the eigenvectors of $S_{XX}$ (loadings)

Square of the singular values are the eigenvalues

Columns of U are the principal components (scores)

# PCA in Regression

## Latent variables, again

Response Y is related to several unobserved latent variable

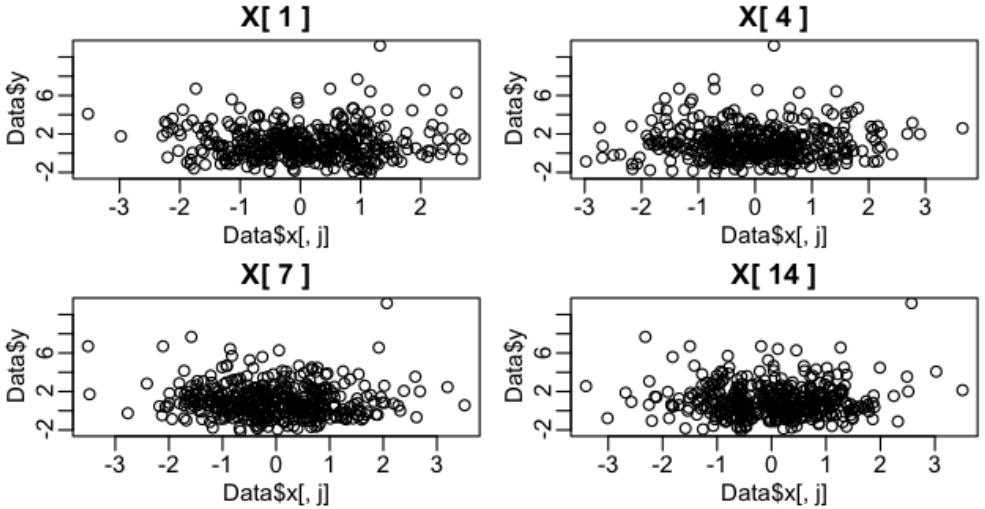
$$Y = \beta_0 + \beta_1 L_1 + \beta_2 L_2 + \ldots + \text{noise}$$

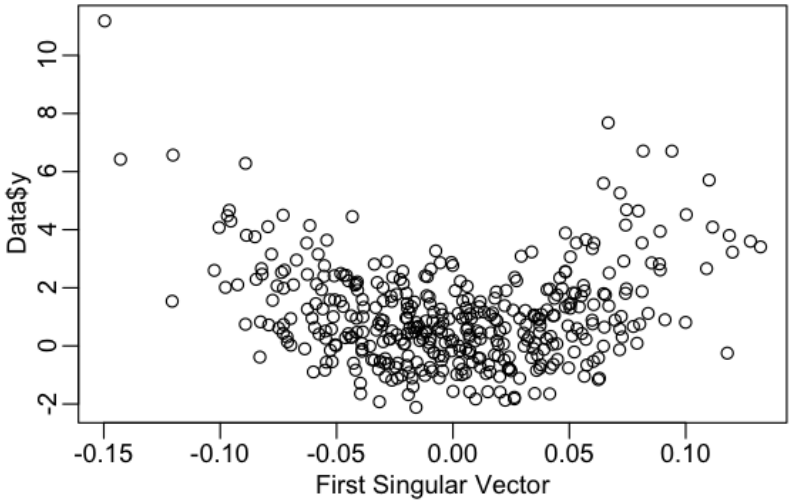
Observe many noisy versions of the latent variables

$$X_j = L_j + \text{random noise}$$

## No evident connect between Y and X's

## Singular vector (PC) reveals nonlinear dependence



# Latent Semantic Analysis

## LSA ≈ PCA of document-term matrix C

## 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)

## Questions

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

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

## Scaling is interesting

### Length normalization

Reduce the influence of longer documents, replacing

$$C_{ij} \rightarrow C_{ij}/n_i \quad \text{or possibly} \quad C_{ij} \rightarrow C_{ij}/\text{sqrt}(n_i)$$

### 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.

$$C_{ij} \rightarrow C_{ij} \times \{\#\ \text{docs}\}/dj$$

### Combinations, such as

$$C_{ij} \rightarrow \log(1+ C_{ij}) \times \log(\{\#\ \text{docs}\}/d_j)$$

# Another Choice: Token Space

## Recall the binary matrices that preserve text information

W indicates word type

D identifies documents

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

documents

| | $d_1$ | $d_2$ | $d_3$ | … | $d_n$ |
|---|---|---|---|---|---|
| $t_1$ | 1 | 0 | 0 | | 0 |
| $t_2$ | 1 | 0 | 0 | | 0 |
| $t_3$ | 1 | 0 | 0 | | 0 |
| $t_4$ | 0 | 1 | 0 | | 0 |
| $t_5$ | 0 | 1 | 0 | | 0 |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| $t_N$ | 0 | 0 | | | 1 |
| | 0 | 0 | | | 1 |

D

word types

| | $w_1$ | $w_2$ | $w_3$ | … | $w_m$ |
|---|---|---|---|---|---|
| $t_1$ | 0 | 0 | 1 | | 0 |
| $t_2$ | 1 | 0 | 0 | | 0 |
| $t_3$ | 0 | 1 | 0 | | 0 |
| $t_4$ | 0 | 1 | 0 | | 0 |
| $t_5$ | 0 | 0 | 0 | | 1 |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| $t_N$ | 0 | 0 | 0 | | 0 |

W

# DTM ≈ Covariance

## Document-type matrix is nxm matrix

$$D^T W = C$$

Counts of the word types in each document

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

## View columns of W and D as indicators

Because most types are rare, means ≈ 0 and

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

## Standardize binomial variation

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

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

$C_{ij} \rightarrow C_{ij}/\mathrm{sqrt}(n_i m_j)$

# Canonical Correlation Analysis

## Extension of regression models 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 D 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., $(X^TX)^{-1}$)

# CCA Calculation

## Matrices X and Y

- Centered so that column means are 0
- Variance/covariances $S_{XX}$ and $S_{YY}$
- Covariance matrix $S_{XY} = S_{YX}^T$

## Classical solution for CCA coefficients

- Eigenvectors of $S_{YY}^{-1}S_{YX}S_{XX}^{-1}S_{XY}$
- Eigenvalues are the squared canonical correlations

## Modern approach

- Singular value decomposition of standardized covariances

# LSA of Wines

wine_spectral.R

# Calculations

## Can you do an LSA?

In wine example, the reduced document term matrix after collecting OOVs is a 20,508 x 2,645 matrix

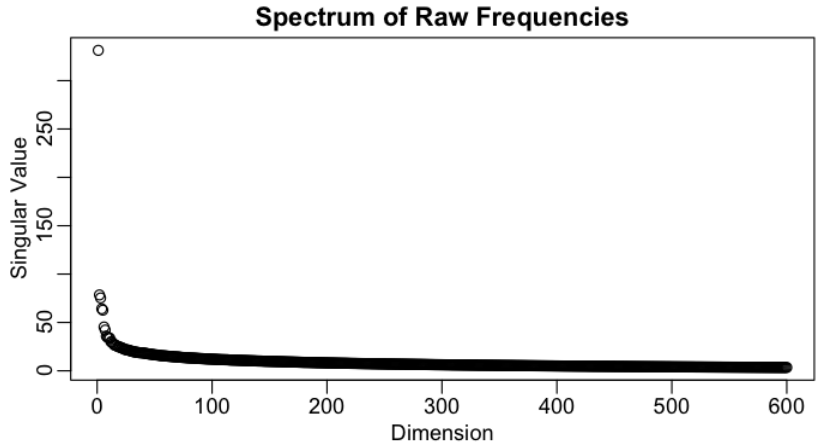
Sparse form saves room, but most SVD code cannot exploit

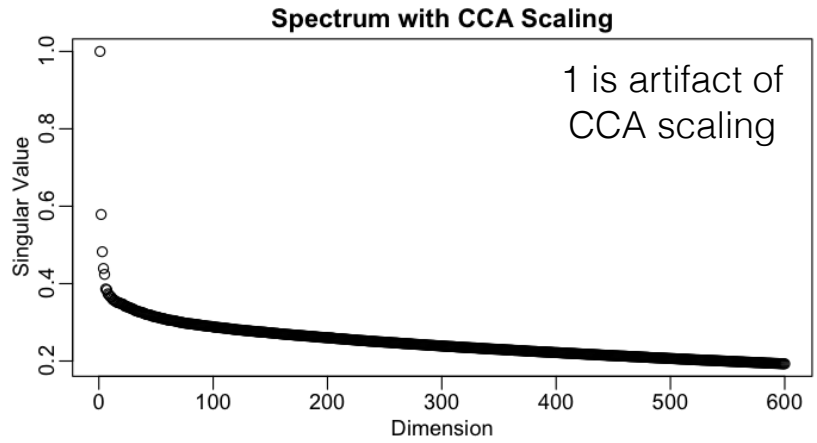
R takes a “long time” to do this calculation

Sample 3,000 documents for illustration

We’ll see how to use the full matrix shortly

## Spectrum



# Calculations

## Can you do an LSA?

In wine example, the reduced document term matrix after collecting OOVs is a 20,508 x 2,659 matrix

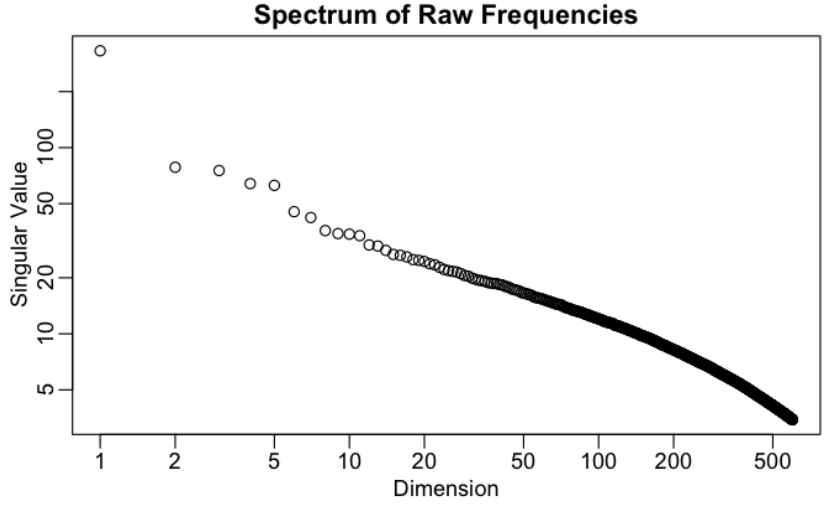
Sparse save room, but most SVD code cannot take advantage

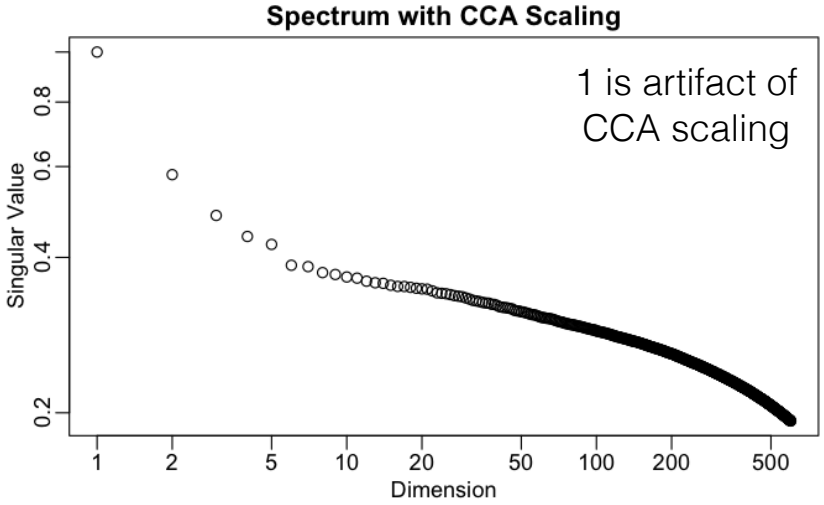
R takes a “long time” to do this calculation

Sample 3,000 documents for illustration

We’ll see how to use the full matrix shortly

## Spectrum (log scale)



# Comparison of Spectra

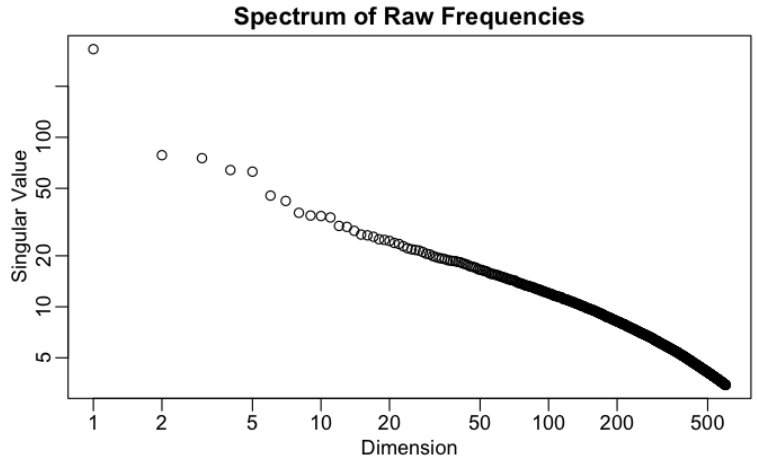
Three versions of frequency weights

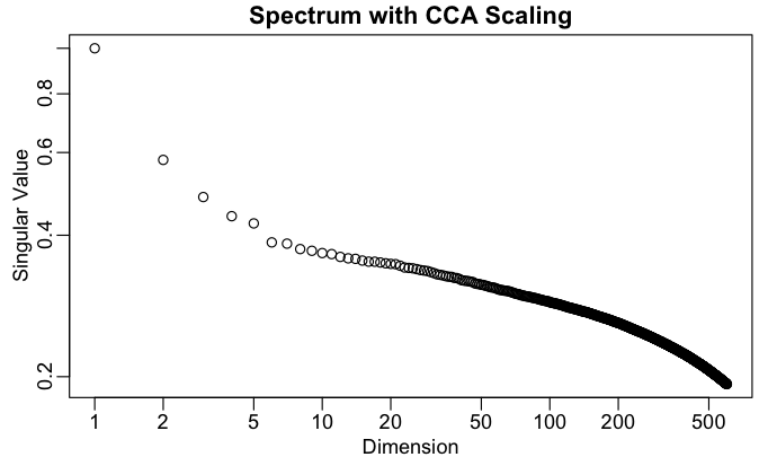
None

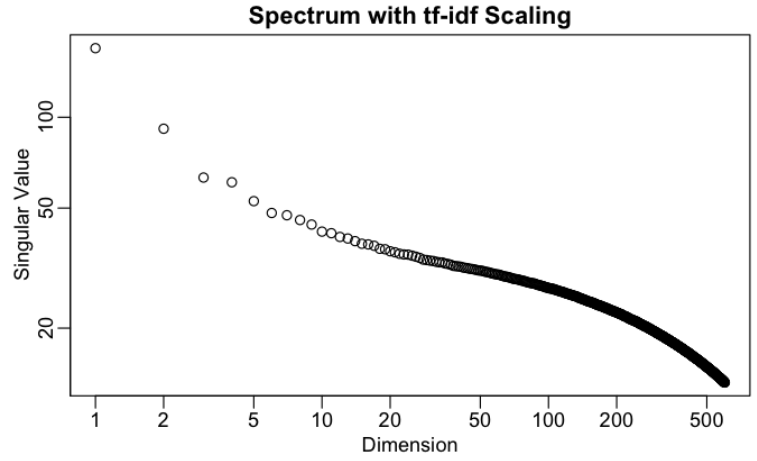
CCA scaling
divide by square root of product of $d_i$ times $m_j$

tf-idf (logs)

Very similar



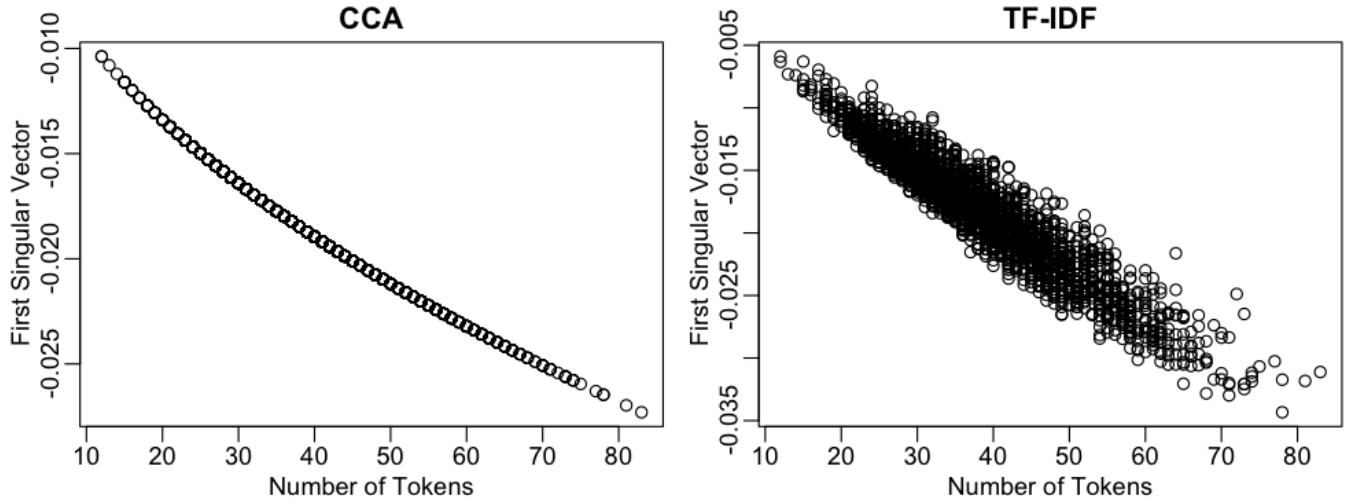



# What are the singular vectors?

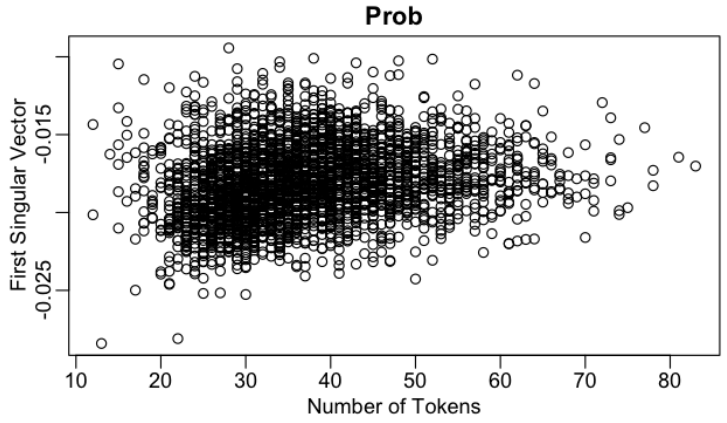
## First is document length

Almost perfect for CCA scaling, highly correlated for others



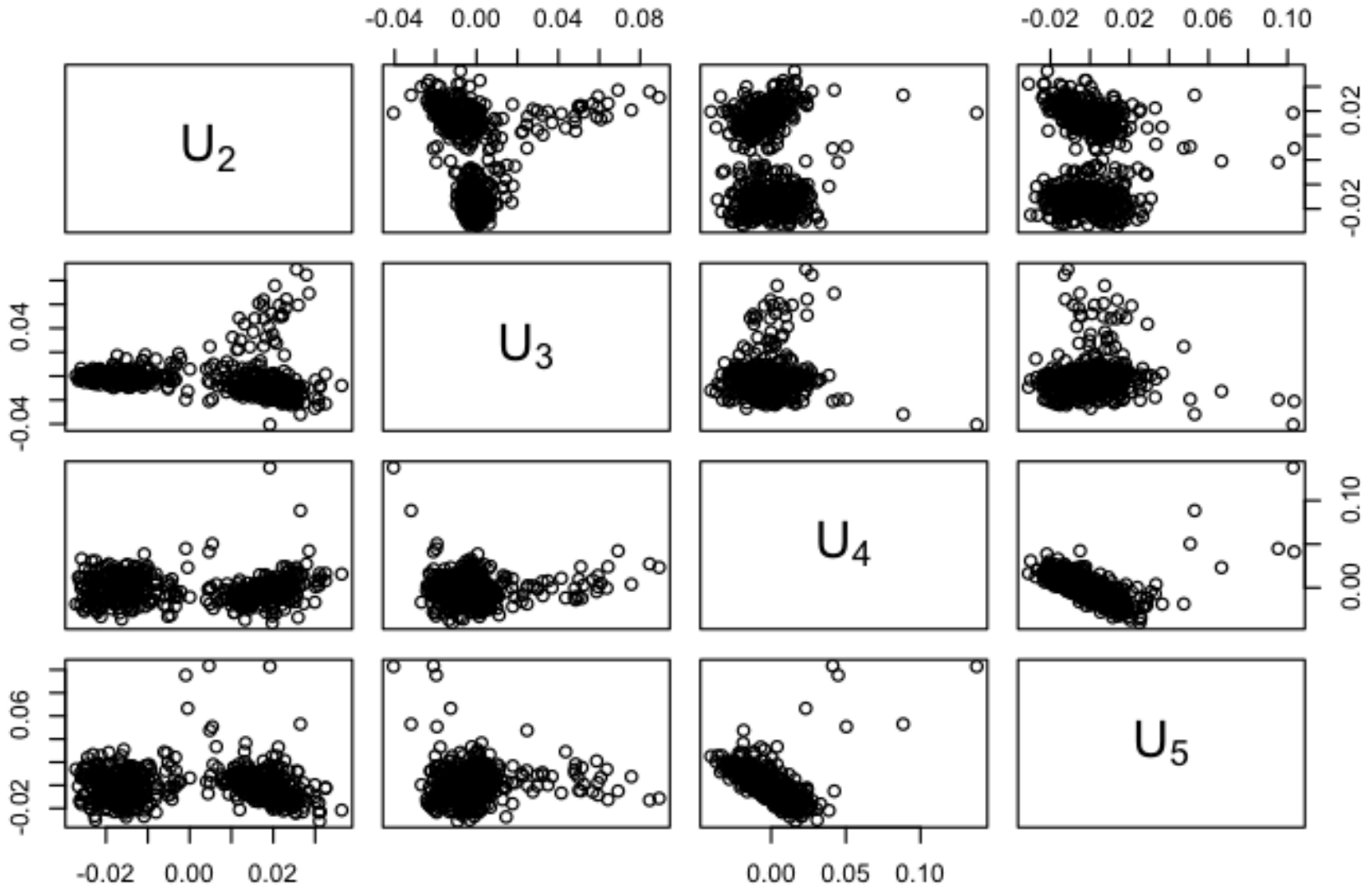

## Can remove by using probability normalization

But do you want to give so much weight to small counts?



# Additional Components

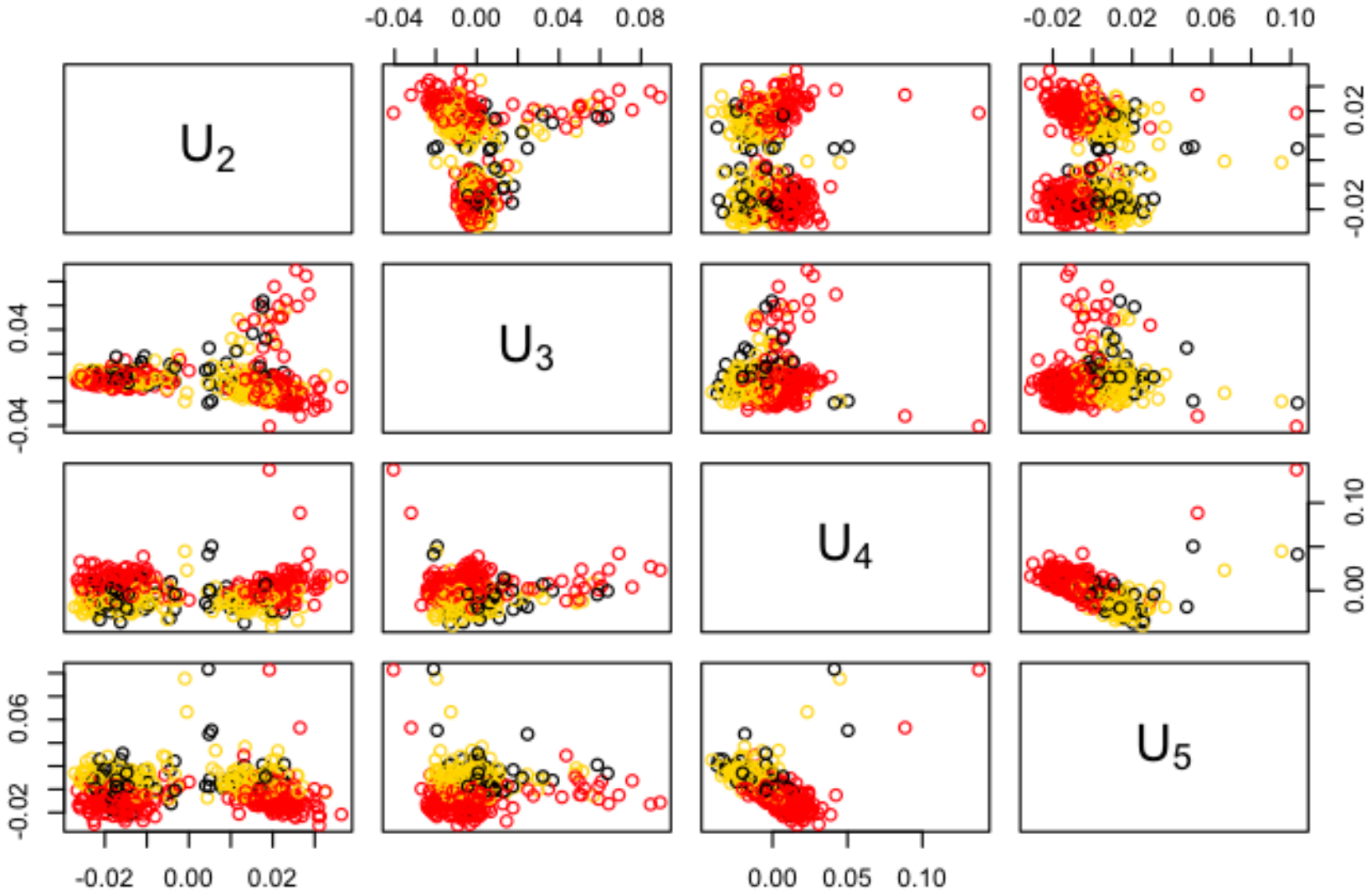
What are the evident clusters?



# Additional Components

Evidently not red and white wines…

you can distinguish red from white, just not the most evident clusters

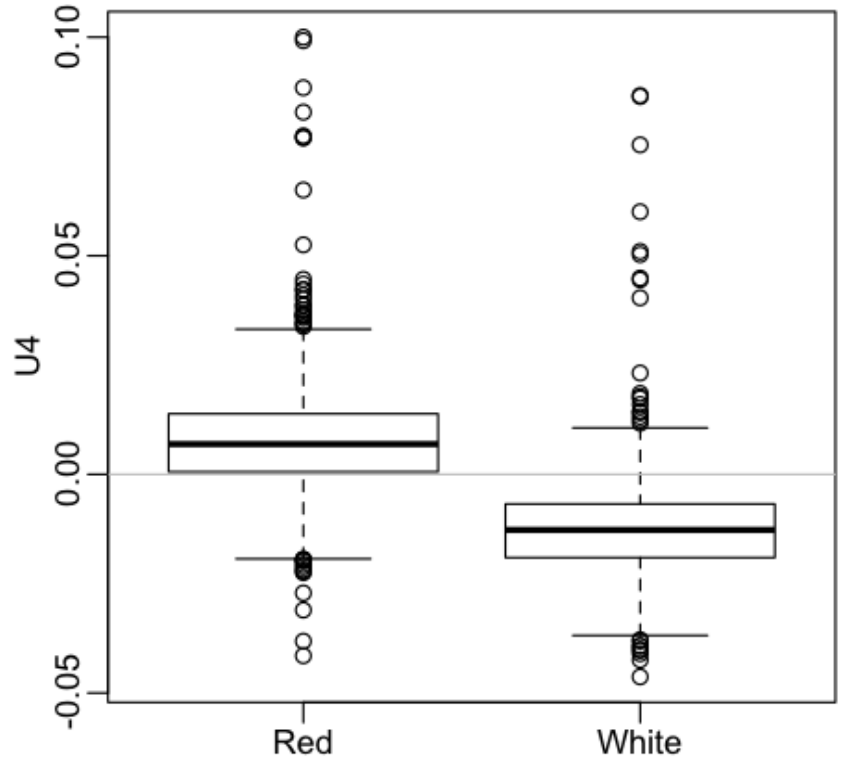


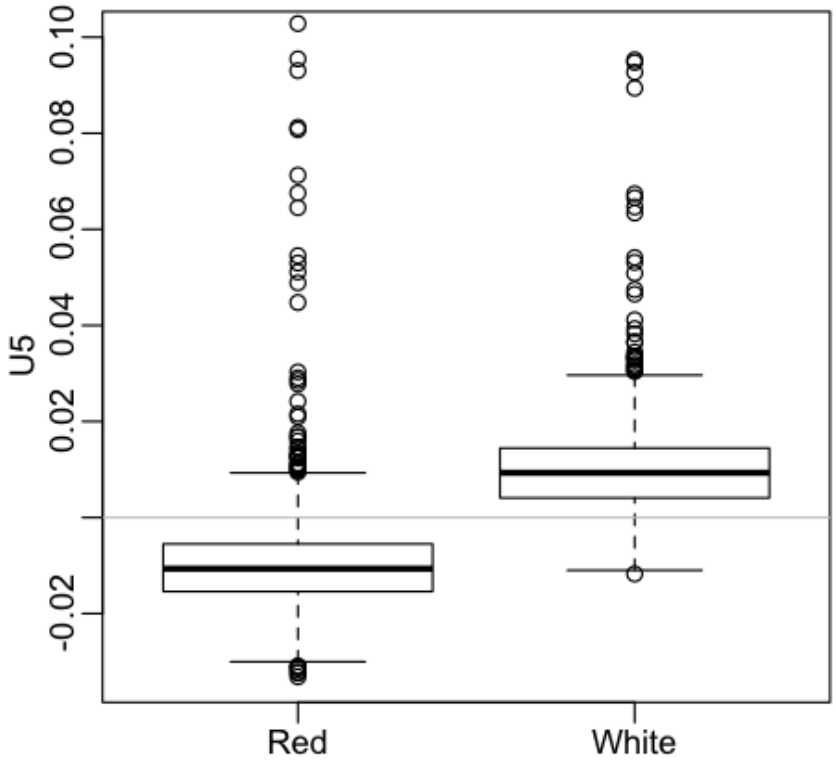
# Color Recognition

Percent Red correct 1442/(85+1442) ≈ 94.4%

Percent White correct 810/(189+810) ≈ 81.1%

Logistic regr did not do much better!



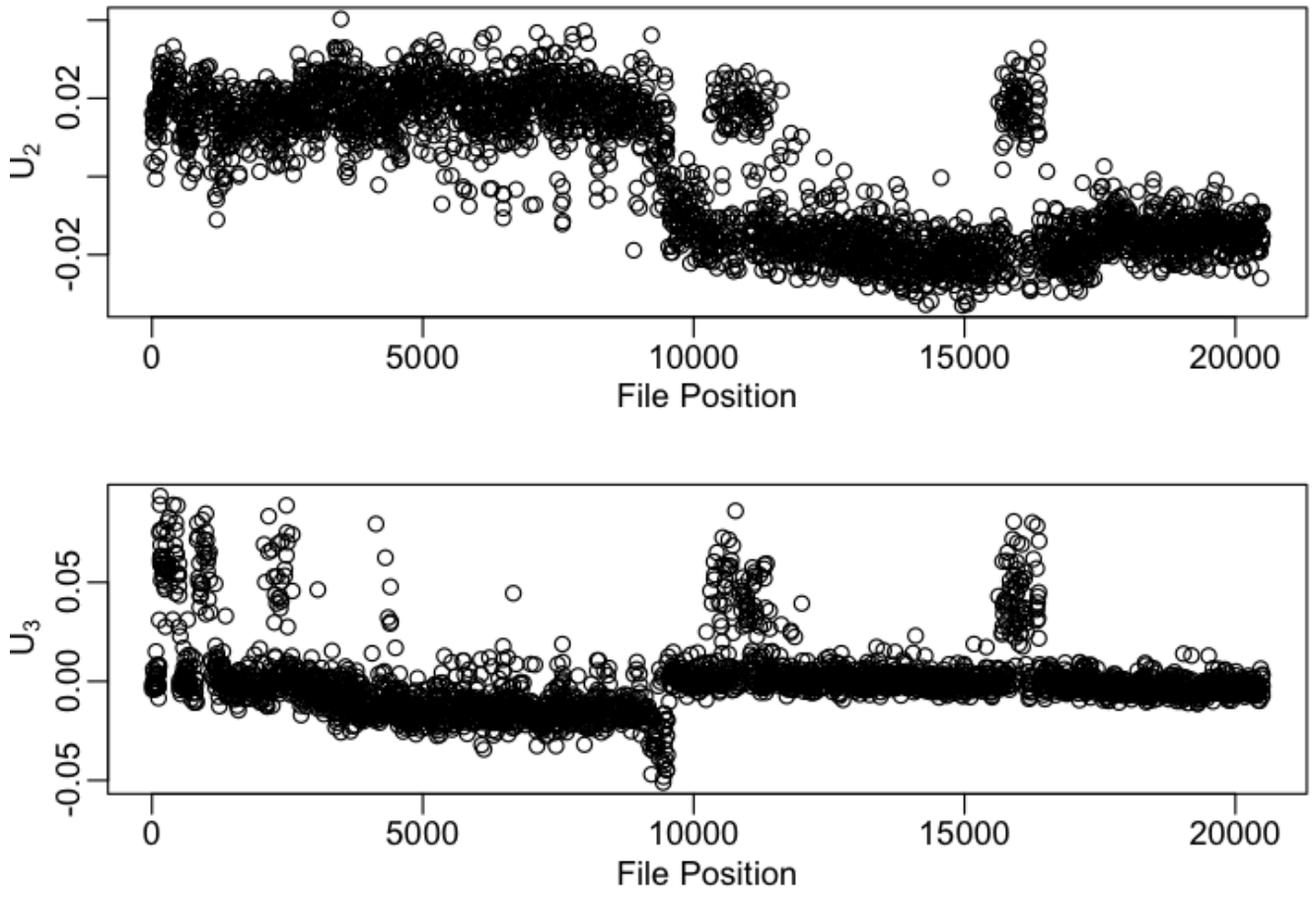

| | FALSE | TRUE |
|---|---|---|
| Red | 1442 | 85 |
| White | 189 | 810 |

# What are the other groups?

## Importance of data familiarity

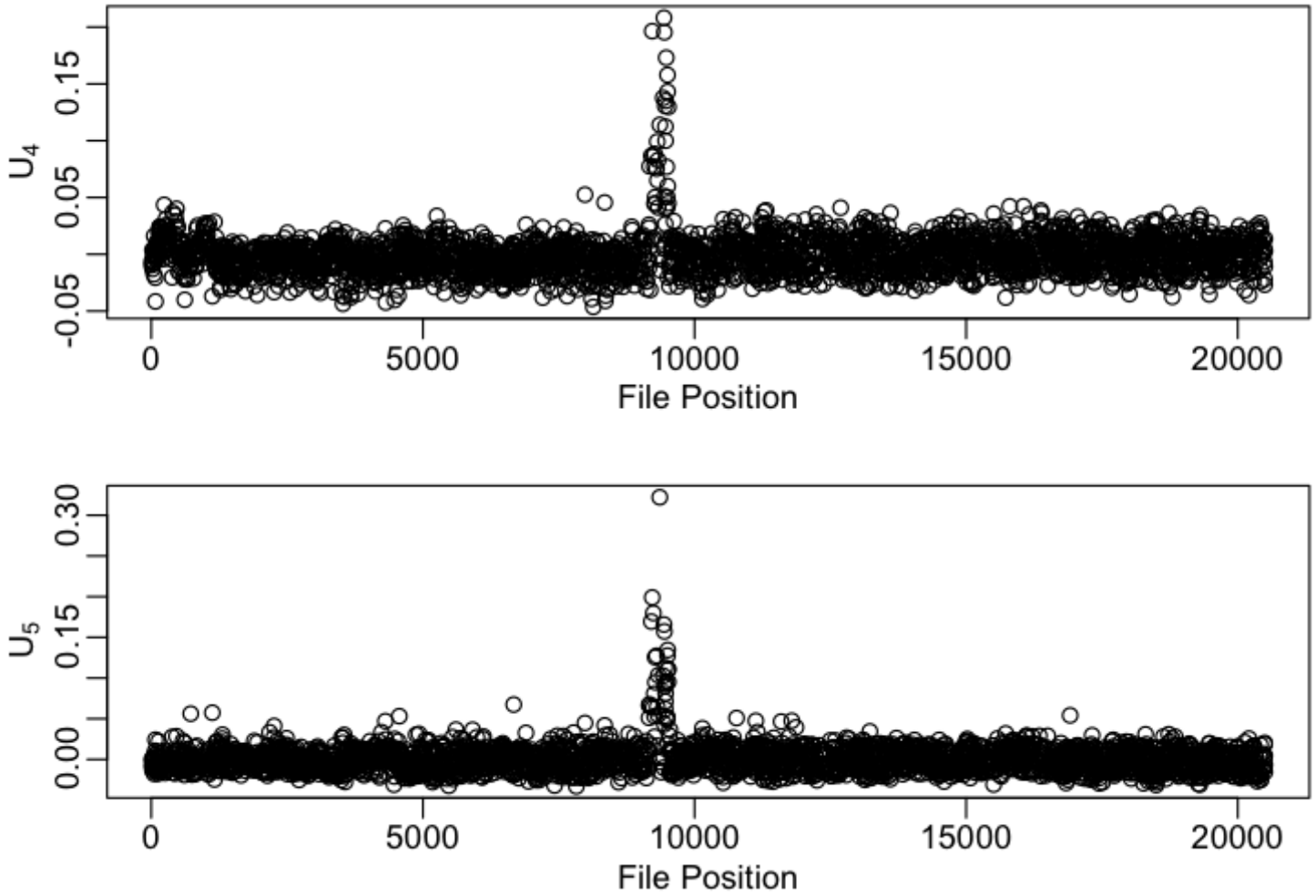
Authorship style of the wine ratings changed over time!

Seldom a bad idea to look for changes over "time"



# And those outliers…

What happened around position 9400?



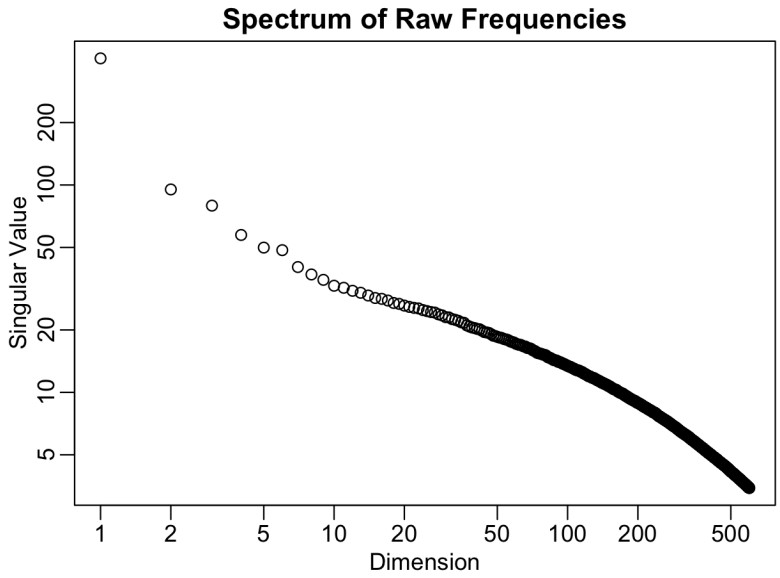

Returning to the original source text reveals what's happening in the underlying text.

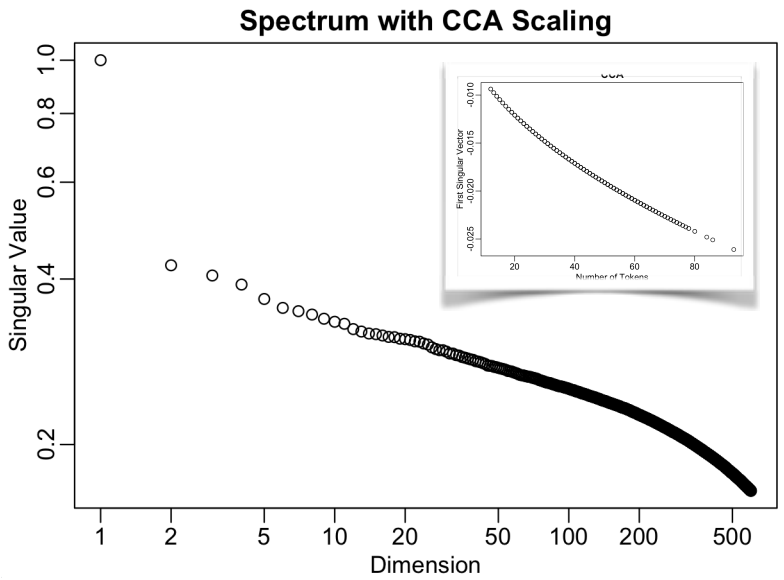
# Analysis: Common Style

## Pick subset of reviews that appear in common style

- Use coordinates of leading prior components
- Reviews typically have food reference

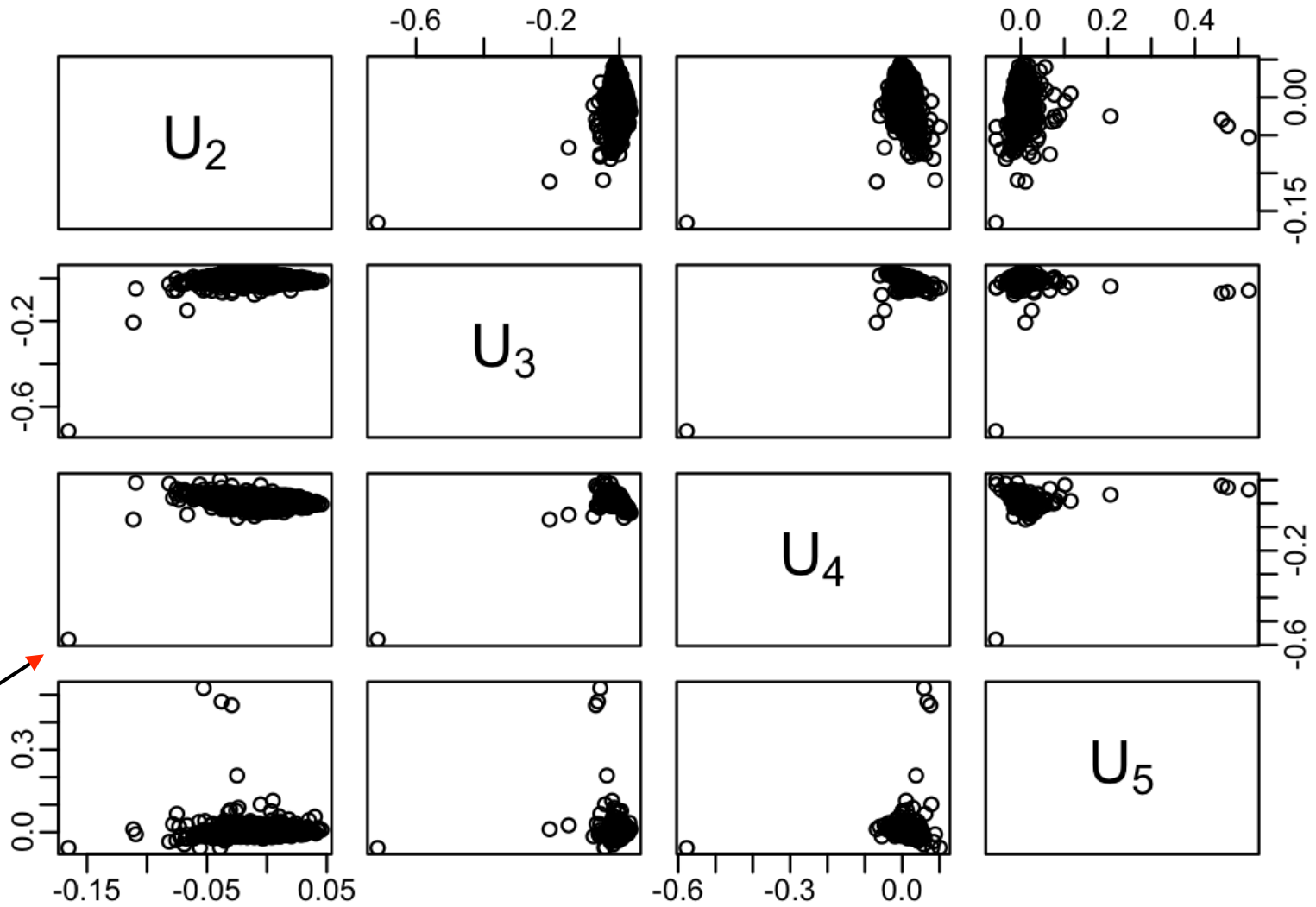
## Spectra resemble those of prior analysis



# Principal Components

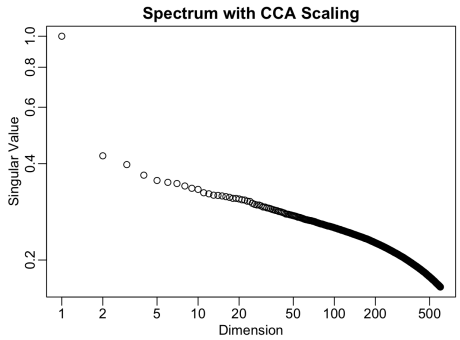
Again find outlying documents (wine reviews)




why so different?

# One more time!

Remove the corked wine and redo the SVD




Spectra is similar, so move on to the components

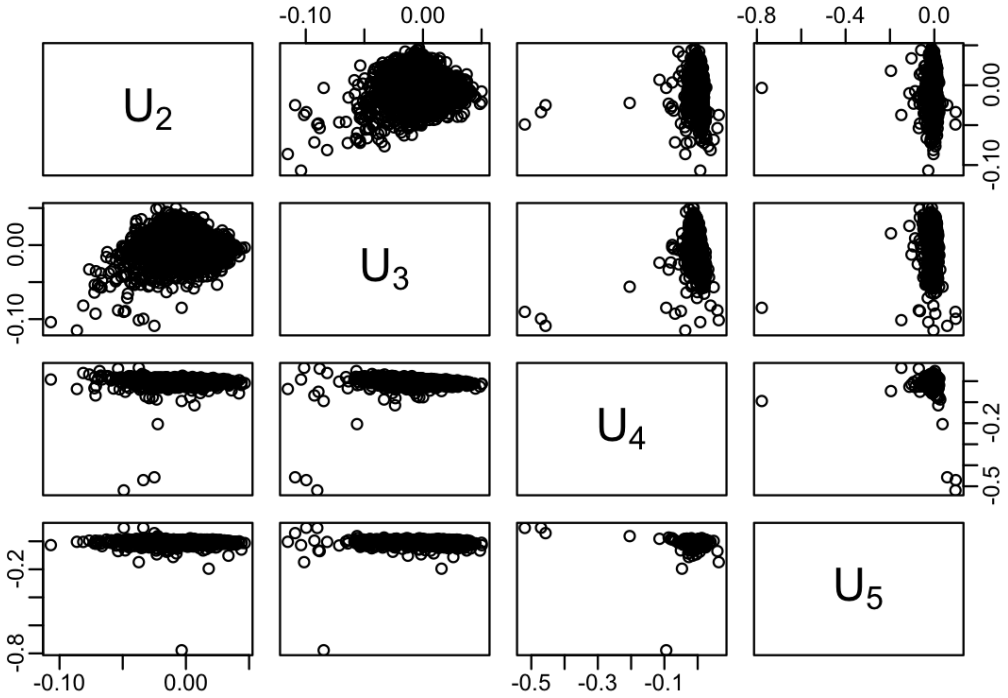
Less clumping of leading component values

Another wacko case!

Description suggests no reason to exclude other than some novel word choices

Remove later outlier

Why some and not others?



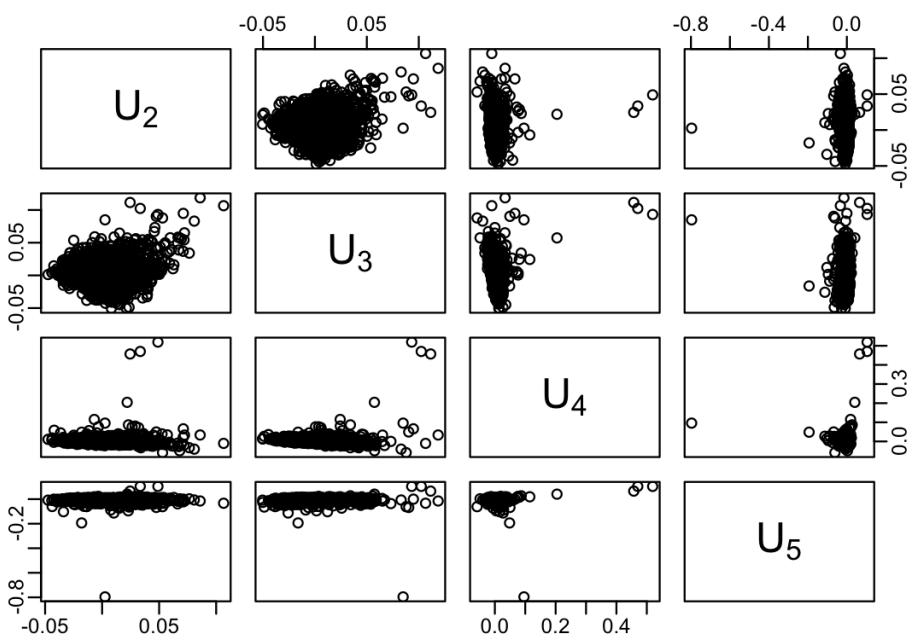
# Moving Along…

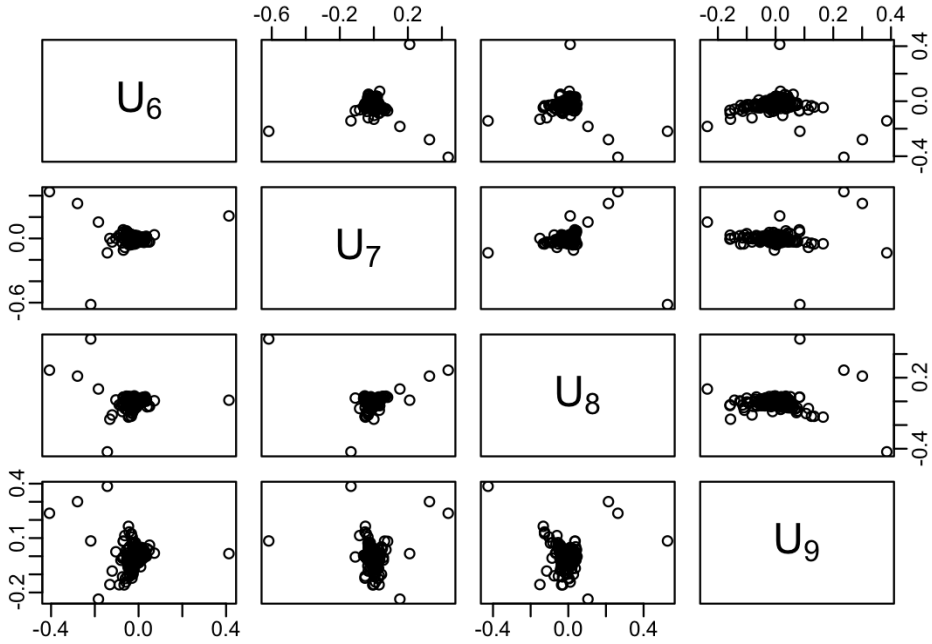
## Reduced to 2,999 cases

Spectrum similar to prior, more outliers remain

None so dramatic as in prior leading components

Some are familiar by now!



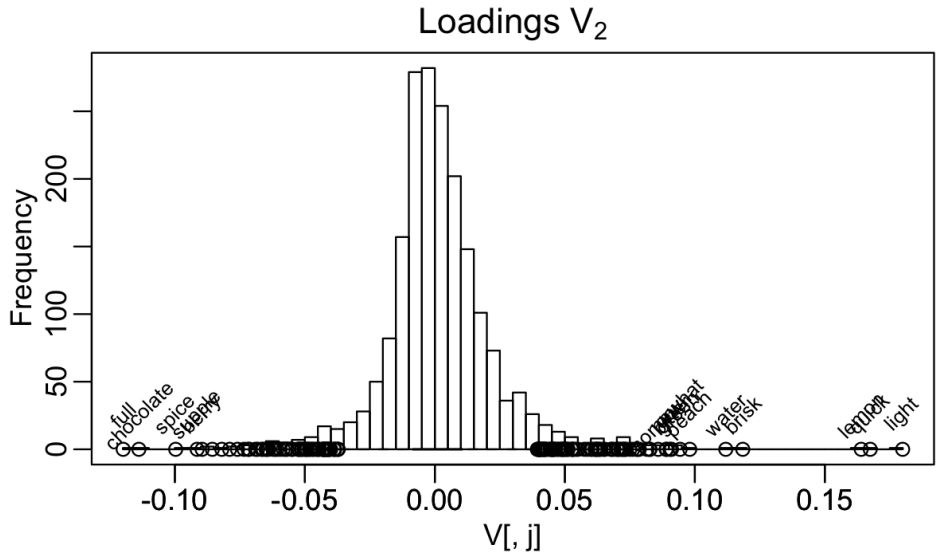



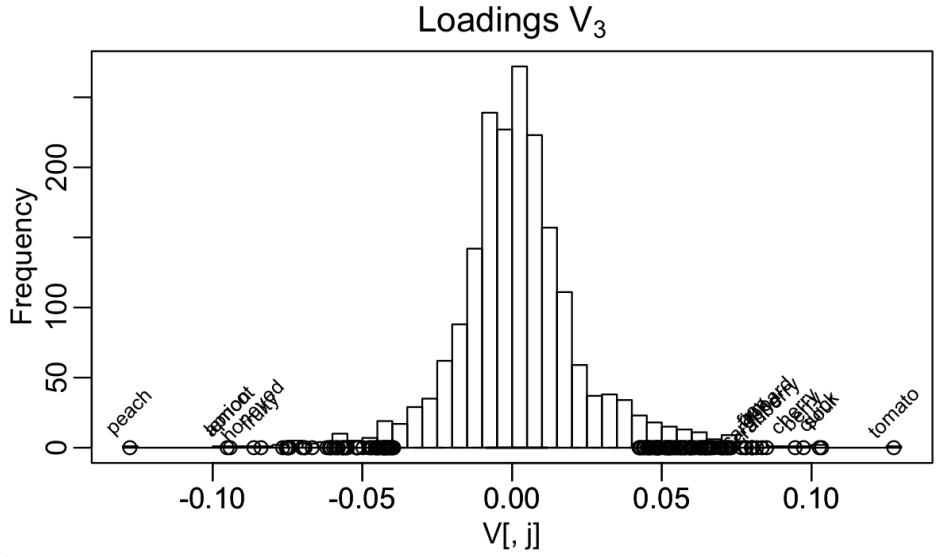
# Which Words

## What are the loadings of the word types?

Coefficients of the left singular vectors, V

Distribution of loadings tend to have long tails

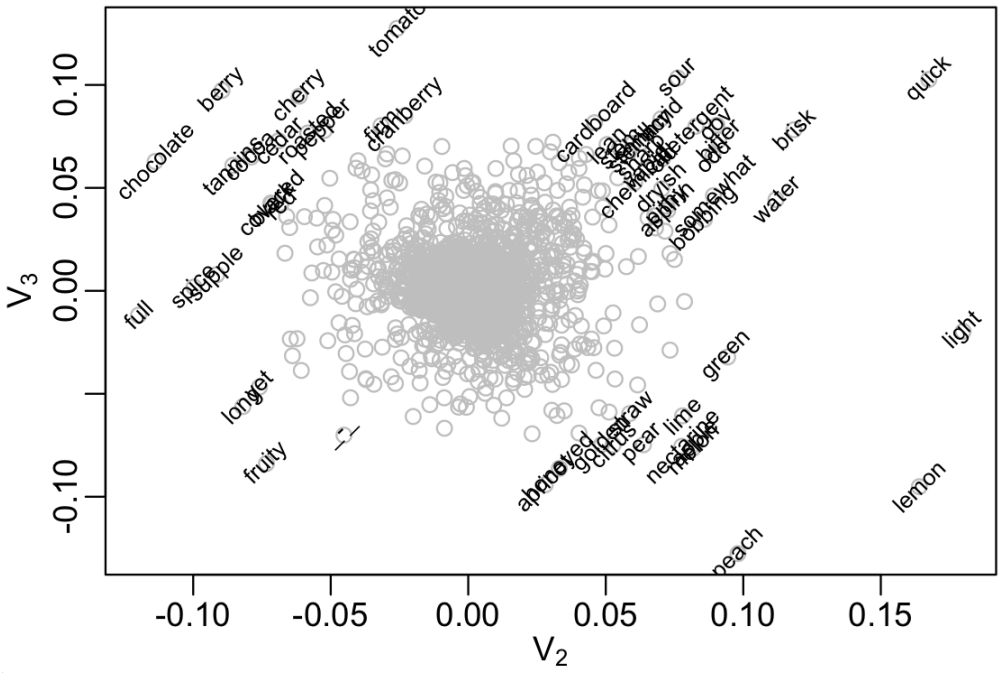


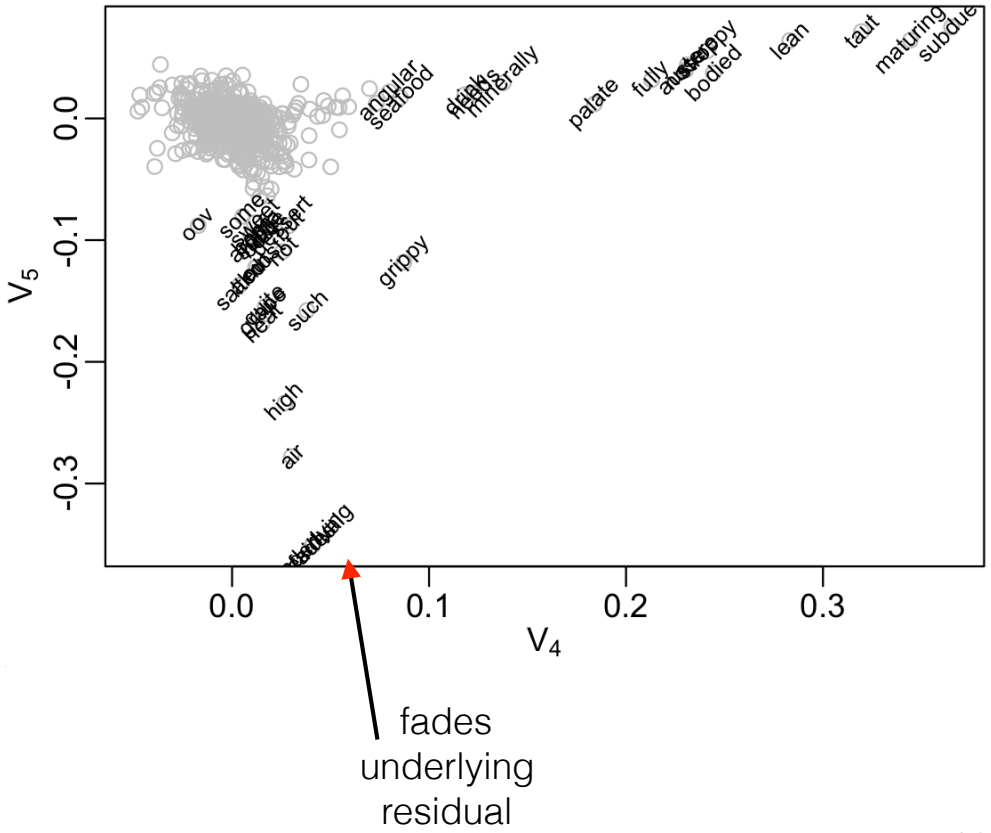



# Which Words

## Scatterplots of word locations

More interesting when not dominated by small subset



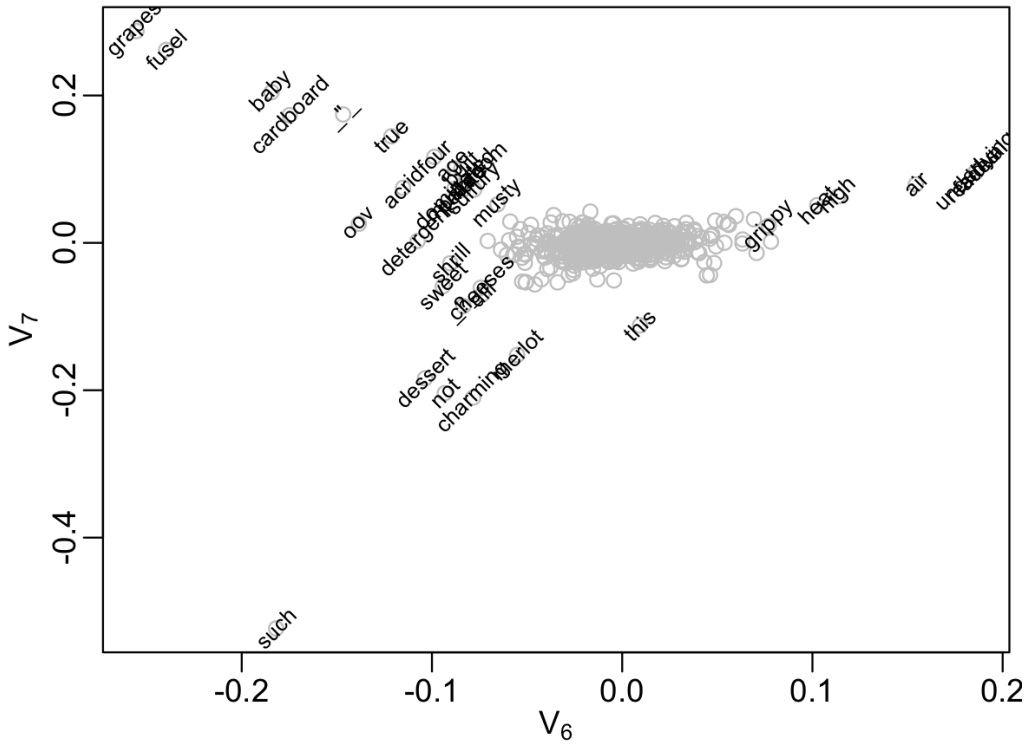
# More Loadings

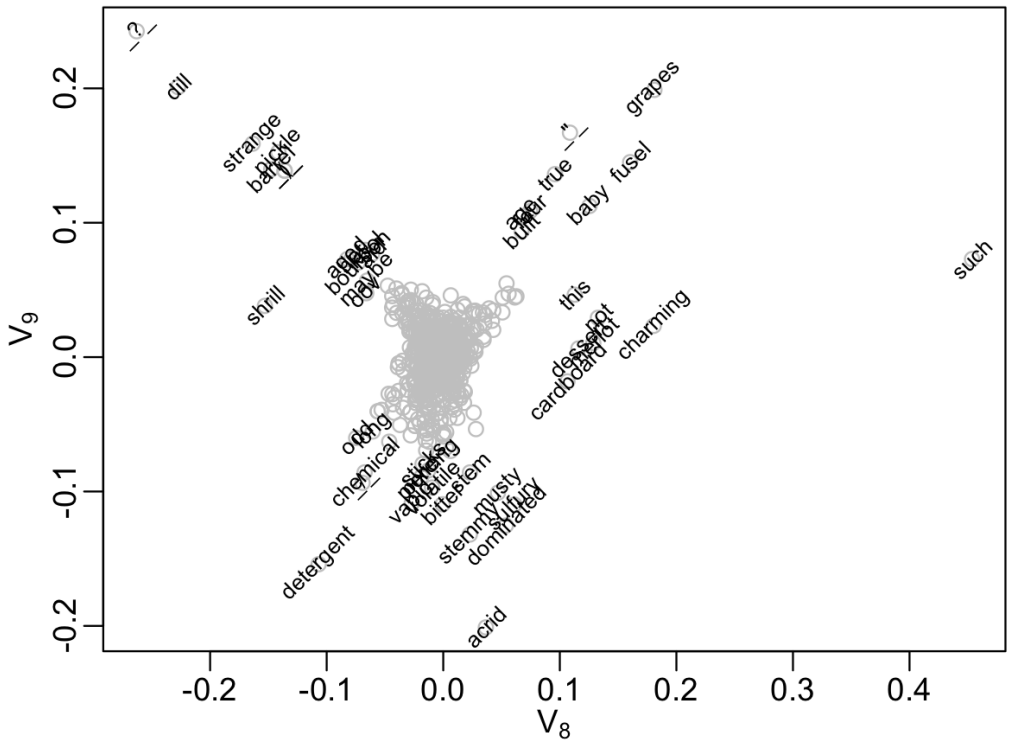
## Concepts are shared over components of SVD

For example, the unpleasant reaction to the wine

Do negative words seem more common?

negative in the sense of wine, that is



# 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

## 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

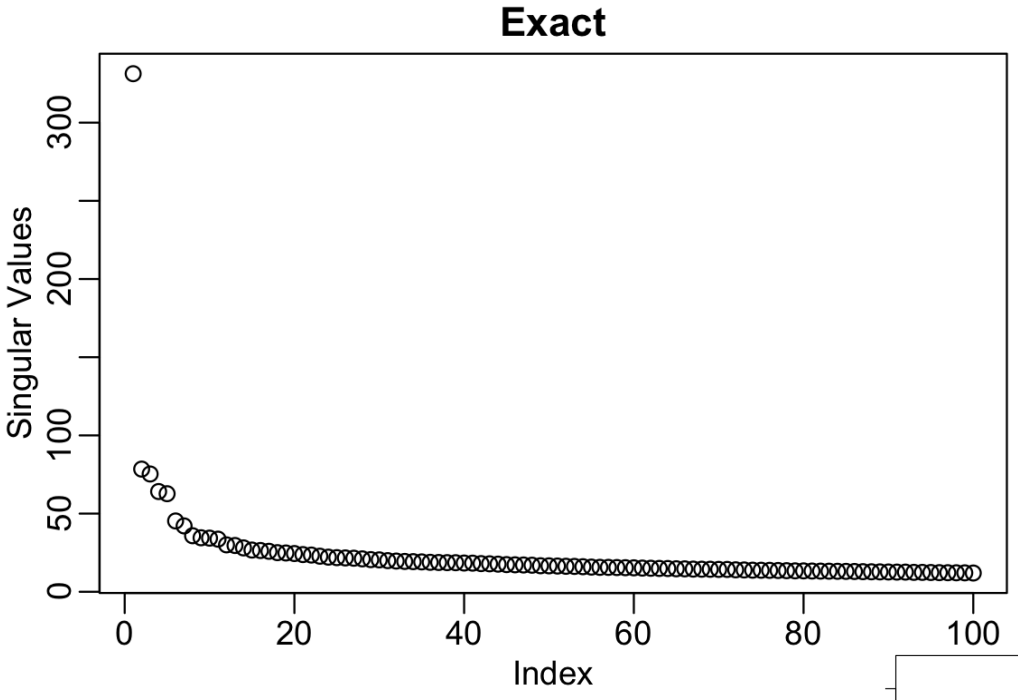
## Algorithm

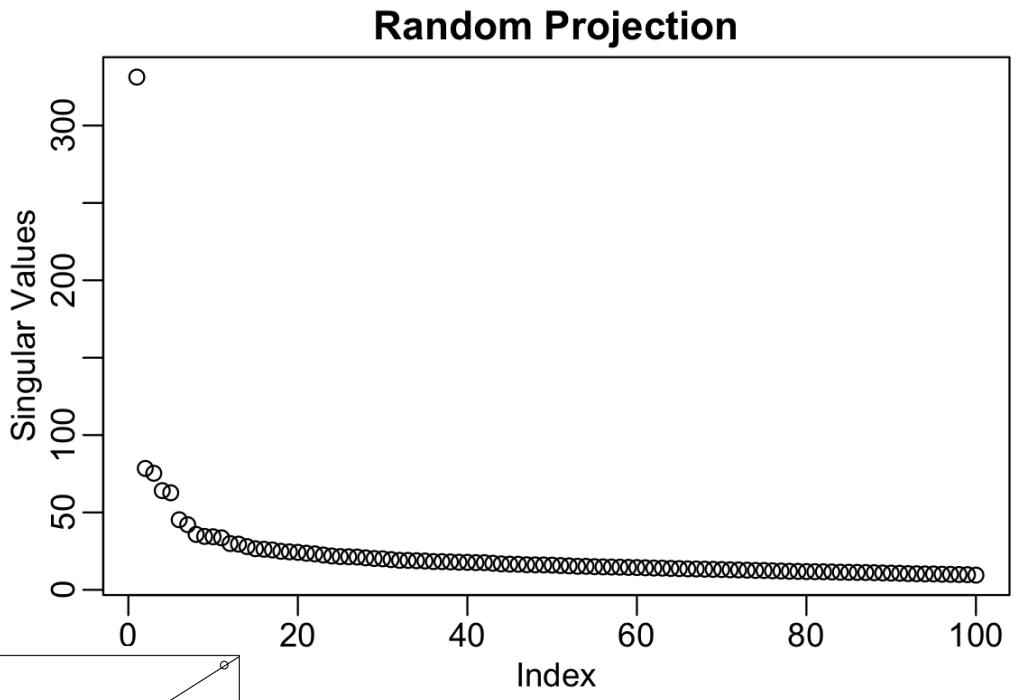
Power iterations improve recovery

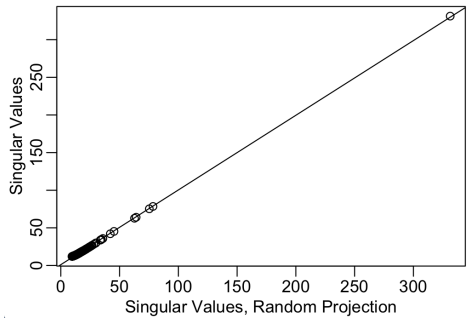
# Comparison to Exact

Compare SVD from random projection to that obtained when R can do the calculation

R can do the SVD of a 3000 x 2250 matrix



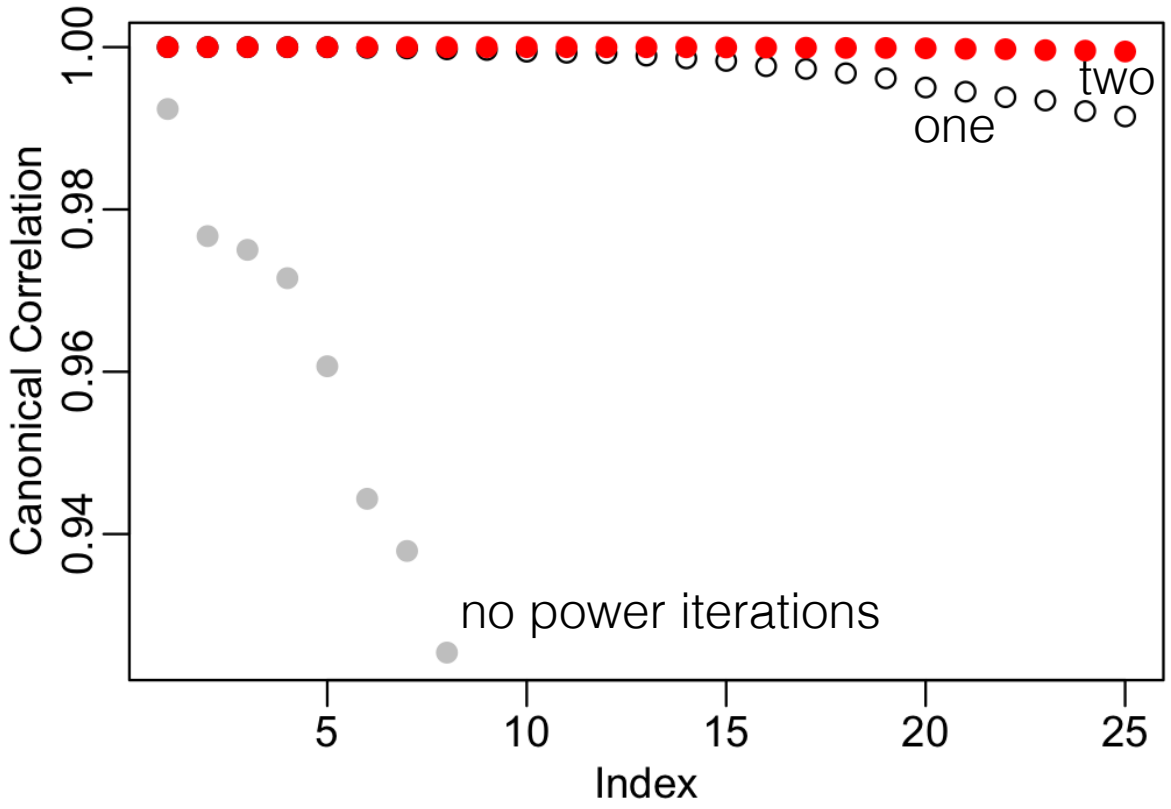



# Subspaces

## What is the relationship between U vectors?

Column span of the resulting singular vectors (components)

Canonical correlations show impact of power iterations



# Principal Components Regression

## Idea

Use random projection to compute PCs of the complete document-term matrix

CCA weighting

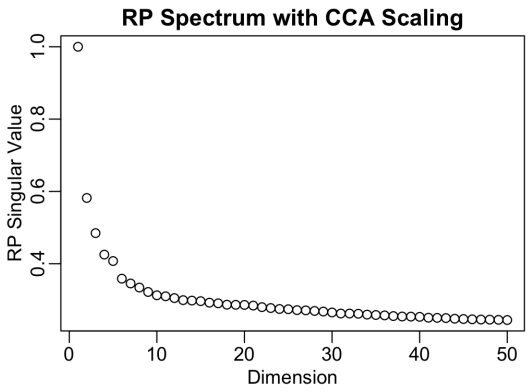
## Transductive framework

Need to compute singular vectors for full data set, both test and train

“Okay” since no response information used in construction

## Random projection

Save 200 columns, with 2 power iterations

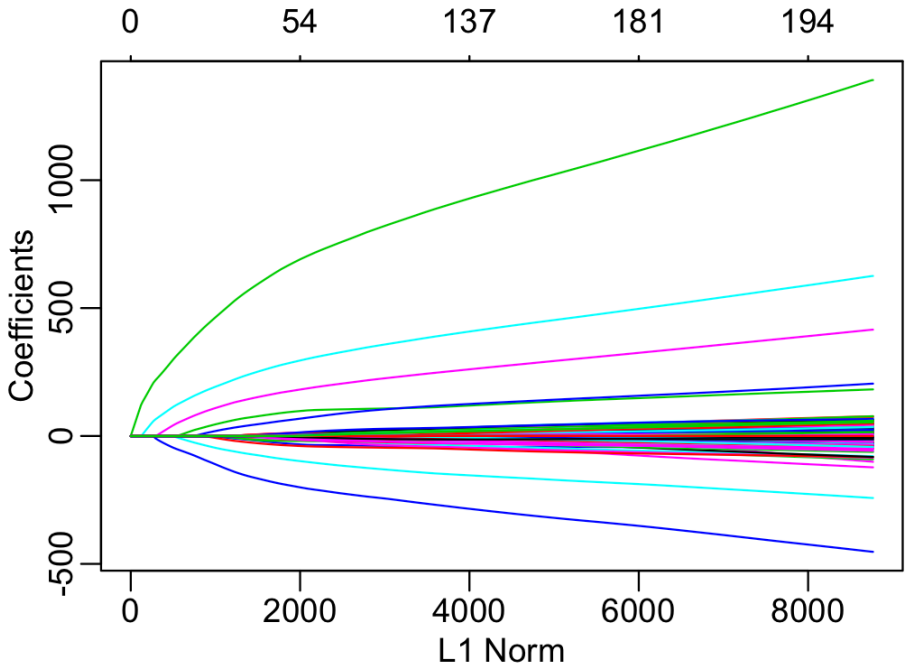


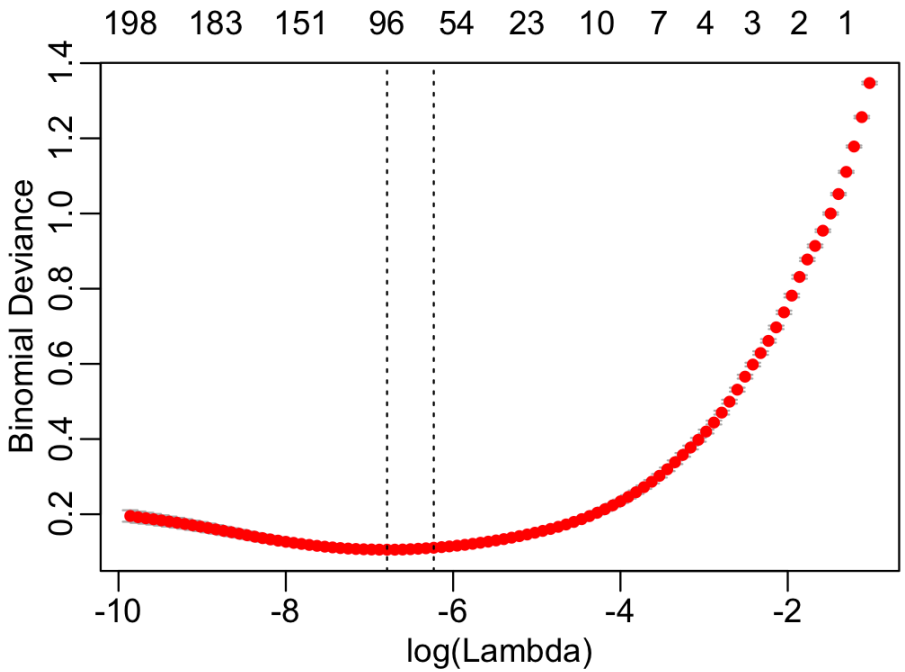
# Lasso with PCs

## Use singular vectors in lasso logistic regression

Only training when fit model

Big effects for $U_4$ - $U_7$



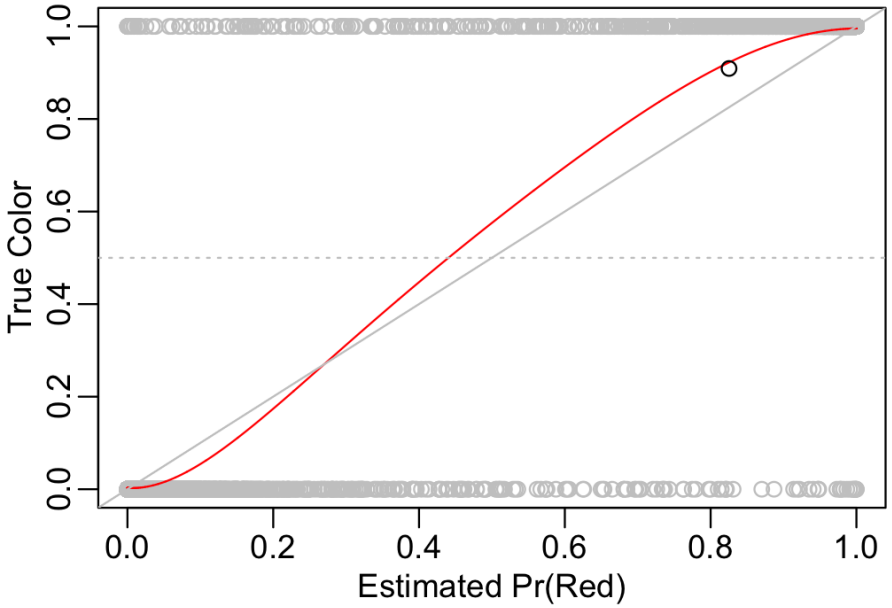

parsimonious “optimal” has 72 coefficients

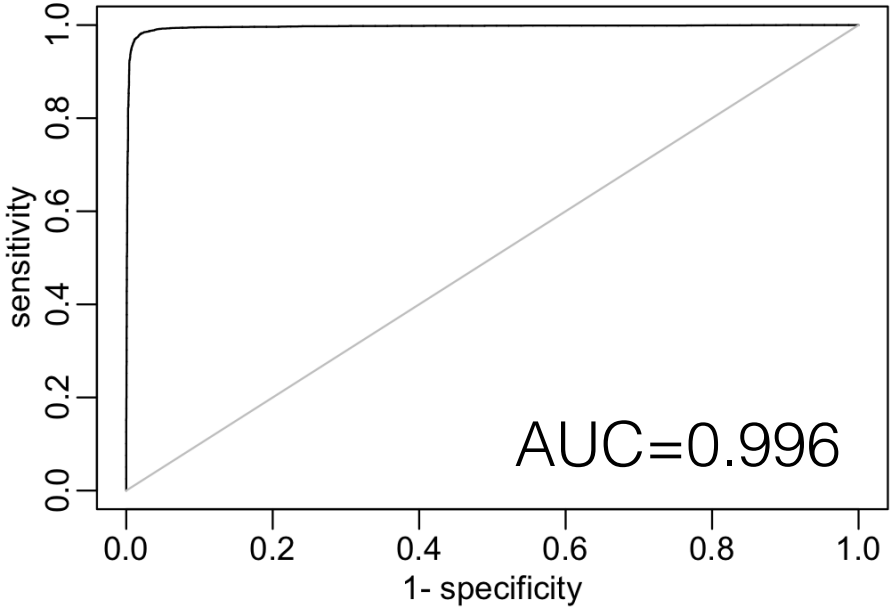
# Classification Results

## Calibration and ROC

Not so well calibrated

Very high AUC



# Confusion Matrix

Very high accuracy, summary measures

| | FALSE | TRUE |
|---|---|---|
| Red | 5955 | 77 |
| White | 118 | 3850 |

White is now “1”

| | Lasso 40 | 200 | PCR |
|---|---|---|---|
| sens | 0.915 | 0.982 | 0.987 |
| spec | 0.891 | 0.987 | 0.970 |
| prec | 0.928 | 0.991 | 0.980 |
| miss | 0.094 | 0.016 | 0.017 |

Using the raw words worked a little better than the PCA/LSA