Untitled

TA: Leslie Huang

Course: Text as Data

Date: 3/29/2018

Recitation 9: Unsupervised Learning I

Set up workspace

```
rm(list = ls())

setwd("/Users/Lingyi/TAD/lab/Text-as-Data-Lab-Spr2018/W9_03_29_18/")

# Loading packages
#install.packages("lsa")
#install.packages("factoextra")

library(quanteda)
```

```
## quanteda version 1.0.0
## Using 3 of 4 threads for parallel computing
##
## Attaching package: 'quanteda'
## The following object is masked from 'package:utils':
##
##     View
```

```
library(quanteda.corpora)
```

1 PCA

1.1 Two functions in base R:

#?prcomp # SVD on the (centered) input data #?princomp # eigendecomposition on the covariance matrix of the input data – can also use option for covariance matrix

Remember to center your data! – use scale() on your matrix beforehand, or the option in prcomp()

And don't have any missing values!

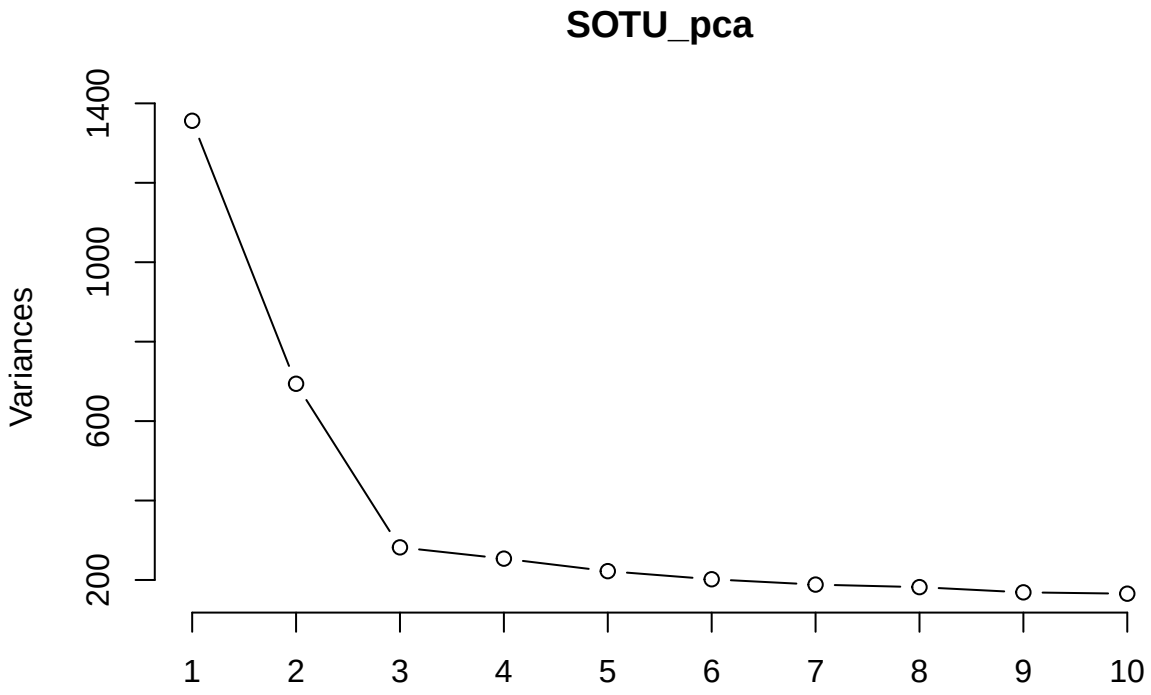
```
library(factoextra) # makes it easy to work with PCA
```

```
## Loading required package: ggplot2
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ
```

1.2 Example

```
data("data_corpus_sotu")

SOTU_dfm <- dfm(data_corpus_sotu[145:223,],
                stem = T,
                remove_punct = T,
                remove = stopwords("english")
)

SOTU_mat <- convert(SOTU_dfm, to = "matrix") # convert to matrix

SOTU_pca <- prcomp(SOTU_mat, center = TRUE, scale = TRUE)

# Elbow plot
plot(SOTU_pca, type = "l")
```



```r
# How much variance do the first few PCs account for?
summary(SOTU_pca)
```

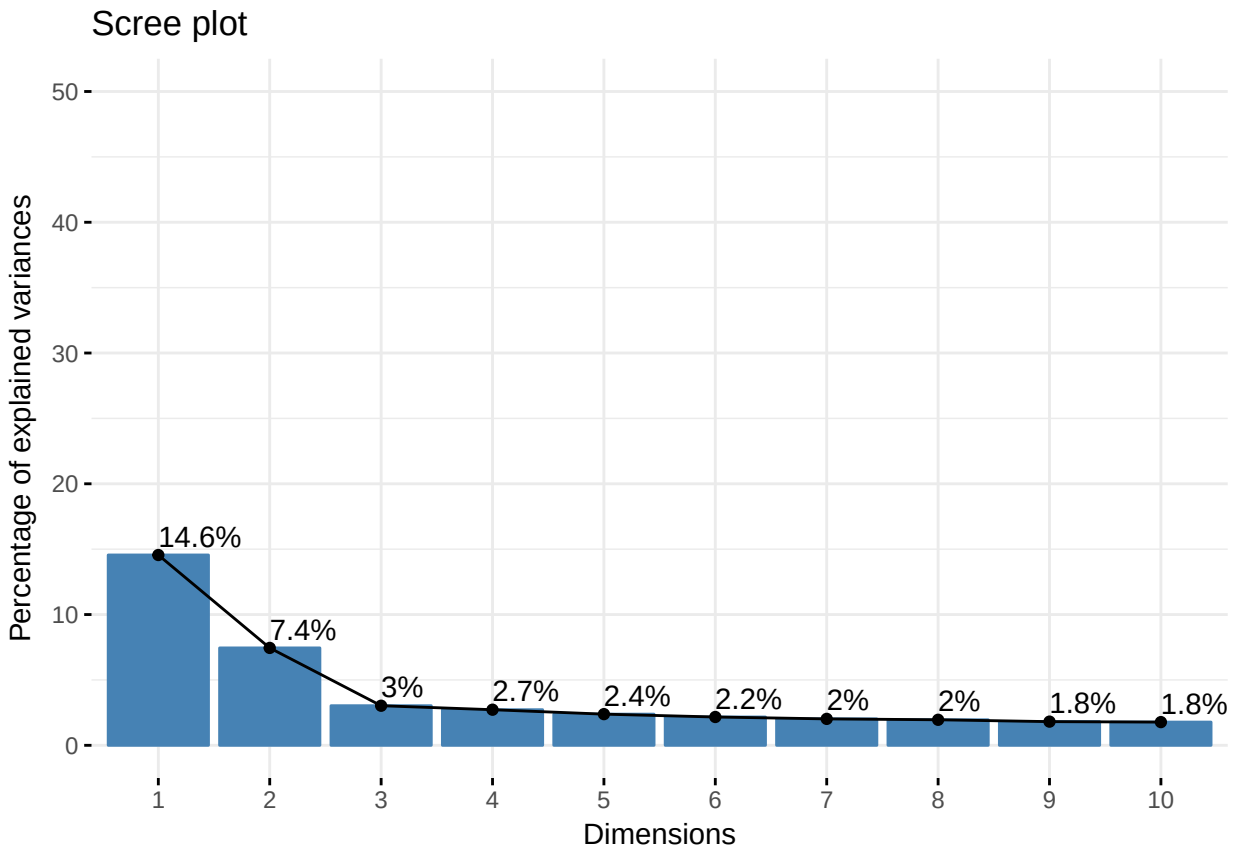
```
## Importance of components:
##                             PC1      PC2      PC3      PC4      PC5
## Standard deviation     36.8254 26.34384 16.79683 15.92878 14.90630
## Proportion of Variance  0.1455  0.07448  0.03028  0.02723  0.02385
## Cumulative Proportion   0.1455  0.22002  0.25029  0.27752  0.30137
##                             PC6      PC7      PC8      PC9     PC10
## Standard deviation     14.20671 13.72426 13.49055 12.99407 12.86534
## Proportion of Variance  0.02166  0.02021  0.01953  0.01812  0.01776
## Cumulative Proportion   0.32303  0.34324  0.36278  0.38090  0.39866
##                            PC11     PC12     PC13     PC14     PC15
## Standard deviation     12.41030 12.29952 12.11120 12.00899 11.86913
## Proportion of Variance  0.01653  0.01624  0.01574  0.01548  0.01512
## Cumulative Proportion   0.41519  0.43142  0.44717  0.46264  0.47776
##                            PC16     PC17     PC18     PC19     PC20
## Standard deviation     11.81085 11.79016 11.76253 11.68813 11.63298
## Proportion of Variance  0.01497  0.01492  0.01485  0.01466  0.01452
## Cumulative Proportion   0.49273  0.50765  0.52250  0.53716  0.55168
##                            PC21     PC22     PC23     PC24     PC25
## Standard deviation     11.55933 11.54741 11.23418 11.11967 10.90046
## Proportion of Variance  0.01434  0.01431  0.01354  0.01327  0.01275
## Cumulative Proportion   0.56602  0.58033  0.59388  0.60715  0.61990
##                            PC26     PC27     PC28     PC29     PC30
## Standard deviation     10.72729 10.56643 10.50511 10.43659 10.20401
## Proportion of Variance  0.01235  0.01198  0.01184  0.01169  0.01117
## Cumulative Proportion   0.63225  0.64423  0.65607  0.66776  0.67894
##                            PC31     PC32     PC33    PC34    PC35    PC36
## Standard deviation     10.14246 10.07314 10.02631 9.97112 9.90171 9.83709
## Proportion of Variance  0.01104  0.01089  0.01079 0.01067 0.01052 0.01039
## Cumulative Proportion   0.68998  0.70087  0.71165 0.72233 0.73285 0.74323
##                           PC37    PC38    PC39    PC40    PC41    PC42
## Standard deviation     9.81327 9.69270 9.66538 9.48507 9.47071 9.41365
## Proportion of Variance 0.01033 0.01008 0.01003 0.00966 0.00963 0.00951
## Cumulative Proportion  0.75357 0.76365 0.77368 0.78333 0.79296 0.80247
##                           PC43    PC44   PC45    PC46    PC47    PC48
## Standard deviation     9.14776 8.96818 8.8994 8.73345 8.62911 8.60696
## Proportion of Variance 0.00898 0.00863 0.0085 0.00819 0.00799 0.00795
## Cumulative Proportion  0.81145 0.82008 0.8286 0.83676 0.84475 0.85271
##                           PC49    PC50    PC51    PC52    PC53    PC54
## Standard deviation     8.50762 8.45944 8.32427 8.29844 8.21210 8.12636
## Proportion of Variance 0.00777 0.00768 0.00744 0.00739 0.00724 0.00709
## Cumulative Proportion  0.86047 0.86815 0.87559 0.88298 0.89022 0.89730
##                           PC55    PC56    PC57    PC58    PC59   PC60
## Standard deviation     7.79589 7.73498 7.42753 7.40955 7.34758 7.2851
## Proportion of Variance 0.00652 0.00642 0.00592 0.00589 0.00579 0.0057
## Cumulative Proportion  0.90383 0.91025 0.91617 0.92206 0.92785 0.9335
##                           PC61    PC62    PC63   PC64   PC65    PC66
## Standard deviation     7.04692 6.92968 6.77176 6.6843 6.6158 6.45239
## Proportion of Variance 0.00533 0.00515 0.00492 0.0048 0.0047 0.00447
## Cumulative Proportion  0.93888 0.94403 0.94895 0.9537 0.9585 0.96291
##                           PC67    PC68    PC69    PC70    PC71   PC72
## Standard deviation     6.37515 6.24832 6.10982 6.09061 5.80466 5.6292
```

```
## Proportion of Variance 0.00436 0.00419 0.00401 0.00398 0.00362 0.0034
## Cumulative Proportion  0.96728 0.97147 0.97547 0.97945 0.98307 0.9865
##                           PC73    PC74    PC75    PC76    PC77   PC78
## Standard deviation     5.55810 5.48423 4.99023 4.56971 3.70130 2.3704
## Proportion of Variance 0.00332 0.00323 0.00267 0.00224 0.00147 0.0006
## Cumulative Proportion  0.98979 0.99301 0.99569 0.99793 0.99940 1.0000
##                           PC79
## Standard deviation     1.372e-12
## Proportion of Variance 0.000e+00
## Cumulative Proportion  1.000e+00
```

```
# Eigenvalues
head(get_eigenvalue(SOTU_pca))
```

```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1  1356.1100        14.553660                    14.55366
## Dim.2   693.9981         7.447930                    22.00159
## Dim.3   282.1336         3.027834                    25.02942
## Dim.4   253.7260         2.722966                    27.75239
## Dim.5   222.1978         2.384608                    30.13700
## Dim.6   201.8305         2.166028                    32.30303
```

```
fviz_eig(SOTU_pca, addlabels = TRUE, ylim = c(0, 50))
```




```
# Loadings for each variable: columns contain the eigenvectors
SOTU_pca$rotation[1:10, 1:5]
```

```
##                  PC1           PC2           PC3           PC4
## mr       -0.008620406  6.040785e-03  0.0132494518 -0.0081215519
```

```
## presid    0.011007151 -8.574709e-05  0.0007250285  0.0215973739
## speaker  -0.006099118  5.222128e-03  0.0141250664 -0.0061487896
## senat     0.013113248  4.278096e-03  0.0064772119 -0.0051333354
## repres    0.015299670 -9.670750e-03 -0.0040156886  0.0051952717
## congress  0.023871449 -6.871078e-03  0.0033853941 -0.0011301375
## come      0.010620353 -6.817789e-03 -0.0017229612  0.0133135918
## open      0.009258969  4.551938e-03  0.0019403140  0.0043621415
## regular   0.021847615 -9.554072e-03  0.0059161126 -0.0004380975
## session   0.000793076  4.650034e-04  0.0006249822  0.0040628312
##                   PC5
## mr        0.010038490
## presid   -0.012276389
## speaker  -0.004969200
## senat    -0.009685591
## repres    0.004234300
## congress -0.001529768
## come     -0.029417605
## open     -0.013972045
## regular   0.007370837
## session   0.015169292
```

```
# Value of the rotated data: your "new", dimensionality reduced data
View(SOTU_pca$x)
```

Visualization resources:

Tutorial from factoextra author about how to use his package to explore and visualize PCA results: http://www.sthda.com/english/articles/31-principal-component-methods-in-r-practical-guide/112-pca-principal-component-analysis-essentials/

See here for visualizing PCA with the ggbiplot library: https://www.r-bloggers.com/computing-and-visualizing-pca-in-r/

2 Latent Semantic Analysis (LSA) aka Latent Semantic Indexing (LSI)

```
library(lsa)
```

```
## Loading required package: SnowballC
```

Let's keep using the SOTU data from before

2.1 Create LSA weights using TDM

```
SOTU_lsa_auto <- lsa(t(SOTU_dfm))
```

```
# Note: We are *not* doing the local/global weighting in this example!
#?gw_idf
# From lecture
# Local weight function: log(tf_ij + 1)
# global weight function: 1 + sum( (p_ij * log(p_ij) ) / log(n) ) where p_ij = tf_ij/gf_i
```

2.2 Check to see what a good number of dimensions is

```
#?dimcalc_share

# lsa_obj$tk -- truncated matrix Tk from term vector matrix T (constituting left singular vectors from
# svd(matrix)$d -- singular values of svd(matrix)
SOTU_lsa_auto_svd <- svd(SOTU_lsa_auto$tk)$d

dimcalc_share(share = 0.5)(SOTU_lsa_auto_svd)
```

```
## [1] 7
```

```
# By default, share is set to .5; let's try .9
# share = fraction of the sum of the selected singular values over the sum of all singular values
dimcalc_share(share = 0.9)(SOTU_lsa_auto_svd)
```

```
## [1] 13
```

```
# Lecture example uses dims = 5
SOTU_lsa_5 <- lsa(t(SOTU_dfm), 5 )

SOTU_lsa_5_mat <- t(as.textmatrix(SOTU_lsa_5) )
```

2.3 Compare features for a few speeches

```
SOTU_dfm@Dimnames$docs[9]
```

```
## [1] "Roosevelt-1942"
```

```
topfeatures(SOTU_dfm[9,])
```

```
##    war  peopl nation   must  fight   unit  world  shall   year    one
##     31     19     19     18     18     17     16     15     14     14
```

```
# With 5 dims:
sort(SOTU_lsa_5_mat[9,], decreasing=T)[1:10]
```

```
##       war     world    nation      year     peopl      must      free
## 23.146413 22.308548 20.493374 13.354537 12.302793 12.087247 11.222742
##        us       can      peac
## 10.735424 10.508095  9.689789
```

```
# With auto (14) dims:
sort(t(as.textmatrix(SOTU_lsa_auto))[9, ], decreasing = T)[1:10]
```

```
##      war   nation     must    world    peopl      can     forc     peac
## 31.25254 18.46004 15.93679 13.71454 13.35374 12.44615 11.83062 11.39943
```

```
## american       us
## 10.90135 10.77034
```

```
# Another example:
SOTU_dfm@Dimnames$docs[55]
```

```
## [1] "Reagan-1985"
```

```
topfeatures(SOTU_dfm[55,])
```

```
##     year       us     must   govern  freedom      can   defens      new
##       31       20       17       15       15       14       12       11
##     time american
##       11       11
```

```
sort(SOTU_lsa_5_mat[55,], decreasing=T)[1:10]
```

```
##     year   nation american    peopl      can  america     must      new
## 18.53841 16.32828 15.96738 14.60866 14.19338 13.73467 13.68963 11.52312
## congress       us
## 11.49666 10.91562
```

```
sort(t(as.textmatrix(SOTU_lsa_auto))[55, ], decreasing = T)[1:10]
```

```
##     year      can american     must  america    peopl    world   nation
## 22.87094 15.83973 15.74190 15.14140 12.82234 12.46284 11.96698 11.51422
##       us congress
## 11.03995 10.26278
```

```
# associate(): a method to identify words that are most similar to other words using a LSA
#?associate
# uses cosine similarity between input term and other terms

SOTU_lsa_3 <- lsa(t(SOTU_dfm), 3 )

SOTU_lsa_3_mat <- as.textmatrix(SOTU_lsa_3)

china <- associate(SOTU_lsa_3_mat, "china", "cosine", threshold = .7)
china[1:10]
```

```
##  catastroph      strive       entir      seaway  forthright        make
##   0.9999050   0.9998449   0.9996695   0.9996301   0.9995587   0.9995265
##       pacif     simplif longer-term   uncorrect
##   0.9994523   0.9993279   0.9993279   0.9993279
```

```
oil <- associate(SOTU_lsa_3_mat, "oil", "cosine", threshold = .7)
oil[1:10]
```

```
##         bus    reassess     uruguay environment    simplist    poseidon
##   0.9999953   0.9999795   0.9999702   0.9999595   0.9999484   0.9999484
##     warhead    reauthor   school-ag     cleanup
##   0.9999484   0.9999399   0.9999384   0.9999384
```

```
america <- associate(SOTU_lsa_3_mat, "america", "cosine", threshold = .7)
america[1:10]
```

```
##        dead        wave     volatil         els       grate      charli
##   0.9999178   0.9998468   0.9998468   0.9998219   0.9997972   0.9997816
## consequenti     violent         thi       faint
```

```
##   0.9997802   0.9997484   0.9997445   0.9997043
```

```
health <- associate(SOTU_lsa_3_mat, "health", "cosine", threshold = .7)
health[1:10]
```

```
## hemispher   correct   western   largest    outsid voluntari      corp
## 0.9998528 0.9998197 0.9998159 0.9998001 0.9997974 0.9997662 0.9996917
##    tradit     civil    altern
## 0.9996299 0.9995116 0.9995005
```

```
# Keep this in mind when we do topic models!
```

2 WORDFISH

How is it different from other approaches we've used for scaling?

2.1 Read in conservative and labour manifestos (from Recitation 6)

```
setwd("../W6_02_27_18/cons_labour_manifestos")

files <- list.files( full.names=TRUE)
text <- lapply(files, readLines)
```

```
## Warning in FUN(X[[i]], ...): incomplete final line found on './Con1929.txt'
```

```
## Warning in FUN(X[[i]], ...): incomplete final line found on './
## Con1974b.txt'
```

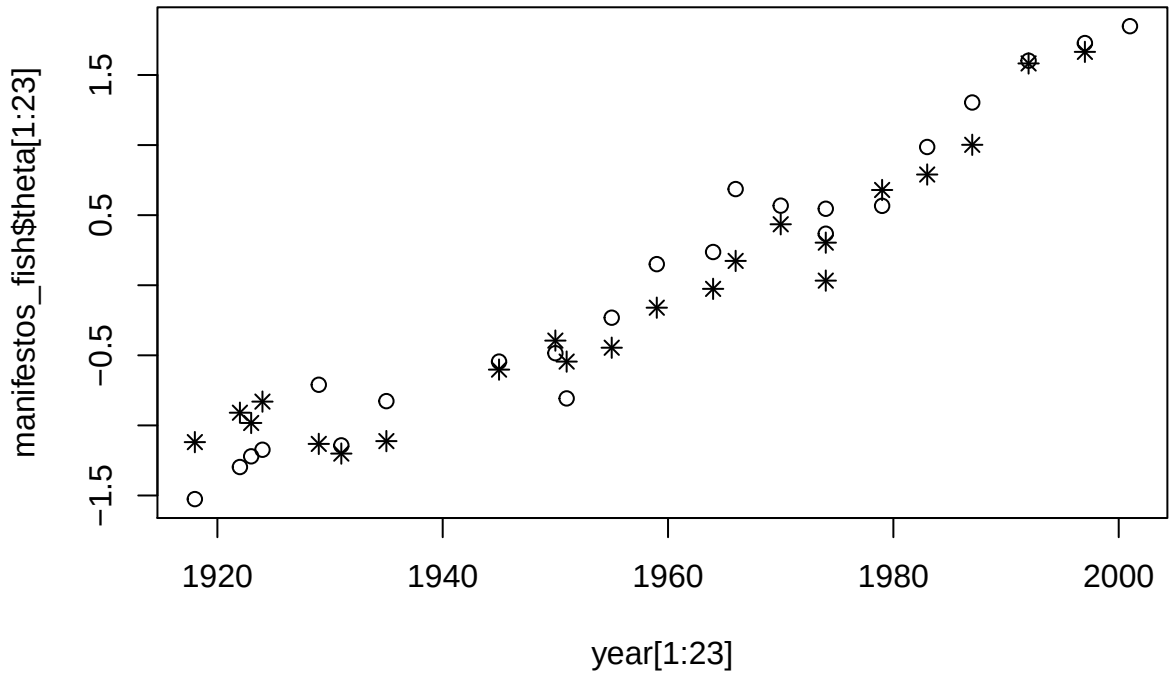
```
## Warning in FUN(X[[i]], ...): incomplete final line found on './Lab2001.txt'
```

```
text<-unlist(lapply(text, function(x) paste(x, collapse = " ")))

# Name data
files <- unlist(files)
files <- gsub("./", "", files )
files <- gsub(".txt", "", files )

# Create metadata
year <- unlist(strsplit(files, "[^0-9]+"))
year <- year[year!=""]

party <- unlist(strsplit(files, "[^A-z]+"))
party <- party[party!="a" & party!="b"]

#create data frame
man_df <- data.frame(year = factor(as.numeric(year)),
                     party = factor(party),
                     text = text,
                     stringsAsFactors = FALSE)

lab_con_dfm <- dfm(man_df$text,
                     stem = T,
```

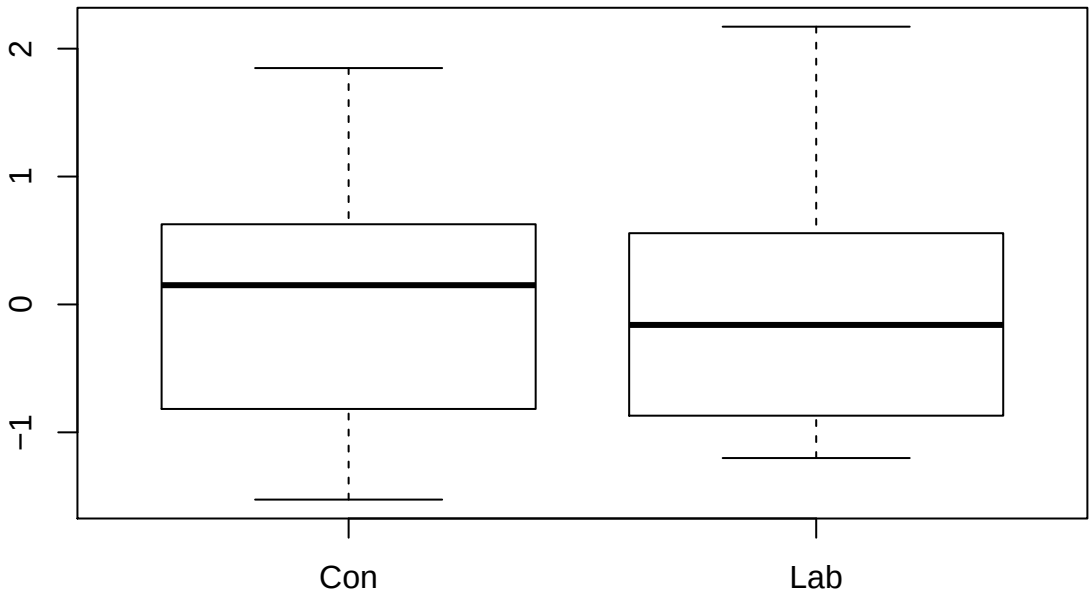
```
                    remove = stopwords("english"),
                    remove_punct = T
                    )
```

2.2 fit wordfish

```
# Setting the index on parties
manifestos_fish <- textmodel_wordfish(lab_con_dfm, c(1,24)) # second parameter corresponds to index tex

# Plot of document positions
plot(year[1:23], manifestos_fish$theta[1:23]) # These are the conservative manifestos
points(year[24:46], manifestos_fish$theta[24:46], pch = 8) # These are the Labour manifestos
```




```
plot(as.factor(party), manifestos_fish$theta)
```




```
# most important features--word fixed effects
words <- manifestos_fish$psi # values
names(words) <- manifestos_fish$features # the words

sort(words)[1:50]
```

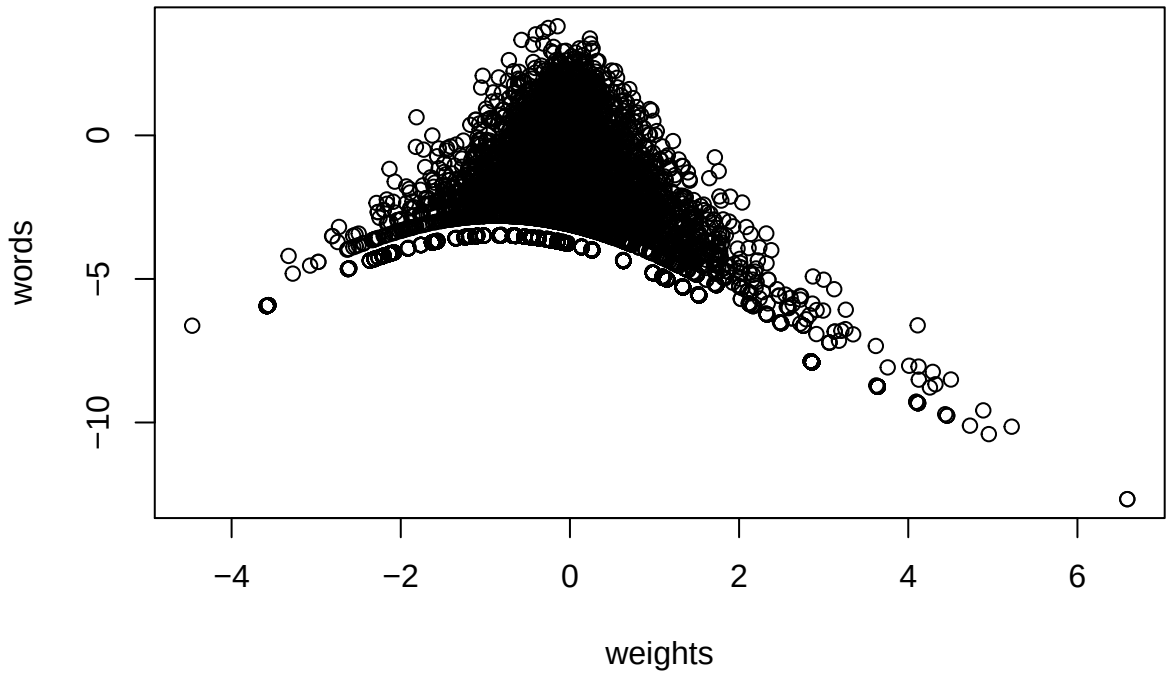
```
##          photo        caption      hard-work           2004           4.20
##     -12.669540     -12.669533     -10.404288     -10.145078     -10.110256
##          genet     step-chang           pcts         on-lin           2003
##      -9.760185      -9.760185      -9.760184      -9.711384      -9.578093
##        newborn three-year-old            1.6            154      supertram
##      -9.326316      -9.326316      -9.326316      -9.326315      -9.326269
##      adulthood             04   middle-incom           rdas          1,200
##      -9.324681      -9.279684      -9.279684      -8.780410      -8.753773
##             ar      good-qual            123         ex-min       paralymp
##      -8.753773      -8.753773      -8.753773      -8.753773      -8.753773
##            csa      demograph       drop-out            sra         seasid
##      -8.753773      -8.753773      -8.753773      -8.753773      -8.753773
##      high-skil         inpati        croydon      sheffield     nottingham
##      -8.753773      -8.753771      -8.753719      -8.753719      -8.753718
##    14-year-old         winner          gcses           tram    11-year-old
##      -8.753682      -8.753607      -8.752032      -8.752030      -8.710004
##           2005    public-priv       frontlin           2010            10p
##      -8.668758      -8.506768      -8.502921      -8.237172      -8.080838
##       internet           2002      in-school           ther      front-lin
##      -8.053294      -8.022851      -7.910242      -7.909388      -7.909387
```

```
sort(words, decreasing=T)[1:50]
```

```
##   govern   labour   nation industri      new    shall   polici    peopl
## 3.800305 3.741035 3.609508 3.518482 3.374958 3.325368 3.197465 3.191267
##     must      can     year     work   servic  britain  increas  countri
## 3.162368 3.081356 3.032595 3.028101 3.028031 2.945773 2.945003 2.931788
##  develop     need     hous   public     make   improv   provid  continu
## 2.929593 2.884721 2.872878 2.792240 2.757552 2.685204 2.644793 2.638296
##    power    parti     plan  british    world    local     help     give
```

```
## 2.626980 2.622481 2.615710 2.611012 2.592590 2.589401 2.588733 2.570085
##      now    trade  conserv    secur   econom   social     educ     home
## 2.562799 2.555097 2.533226 2.519774 2.517468 2.510158 2.494830 2.468264
## programm     also    price    right    first   system     time      act
## 2.457924 2.456635 2.428989 2.402857 2.394542 2.393176 2.366230 2.347593
##  pension   scheme
## 2.341391 2.340782
```

```r
# Guitar plot
weights <- manifestos_fish$beta

plot(weights, words)
```




```r
# also check out wordshoal!
```