Project1

Code

Xiuruo (Sharon) Yan

1/29/2018

Inaugural speeches are the beginning of presidency for every American presidents, which are supposed to cover the heat of time and show the presidents' political position. When there is a recession, would presidents tend to use sepecific words? Would they focus more on economy? Would they feel fear? This report focus on above questions.

According to the list of recessions in the United States (https://en.wikipedia.org/wiki/List_of_recessions_in_the_United_States (https://en.wikipedia.org/wiki/List_of_recessions_in_the_United_States)), I pick 14 presidents who gave their inaugural speeches during recessions. They are Barack Obama, Ronald Reagan, John.F.Kennedy, Harry.S.Truman, William McKinley, Andrew Jackson, Abraham Lincoln, James.K.Polk, Grover Cleveland, William Henry Harrison, John Quincy Adams, James Monroe, Thomas Jefferson, John Adams. In the following report, I use "the first group" to represent these 14 presidents, and "the other group" to represent other presidents.

I will use worldcloud to find presidents' most used words, topic models to find their topics, and sentiment analysis to analyze their emotions.

Step 0: Gather data

Check and install needed packages. Load the libraries and functions.

Hide

Hide

```
packages.used=c("rvest", "tibble", "qdap",
                "sentimentr", "gplots", "dplyr",
                "tm", "syuzhet", "factoextra",
                "beeswarm", "scales", "RColorBrewer",
                "RANN", "tm", "topicmodels")
# check packages that need to be installed.
packages.needed=setdiff(packages.used,
                        intersect(installed.packages()[,1],
                                  packages.used))
# install additional packages
if(length(packages.needed)>0){
  install.packages(packages.needed, dependencies = TRUE)
}
# load packages
library("rvest")
library("tibble")
# You may need to run
# sudo ln -f -s $(/usr/libexec/java_home)/jre/lib/server/libjvm.dylib /usr/local/lib
# in order to load qdap
library("qdap")
library("sentimentr")
library("gplots")
library("dplyr")
library("tm")
library("syuzhet")
library("factoextra")
library("beeswarm")
library("scales")
library("RColorBrewer")
library("RANN")
library("tm")
library("topicmodels")
library("wordcloud")
library(tidytext)
source("../lib/plotstacked.R")
source("../lib/speechFuncs.R")
```

This notebook was prepared with the following environmental settings.

Hide

Hide

```
print(R.version)
```

```
               _
platform       x86_64-apple-darwin15.6.0
arch           x86_64
os             darwin15.6.0
system         x86_64, darwin15.6.0
status
major          3
minor          4.3
year           2017
month          11
day            30
svn rev        73796
language       R
version.string R version 3.4.3 (2017-11-30)
nickname       Kite-Eating Tree
```

Data harvest: scrap speech URLs from http://www.presidency.ucsb.edu/ (http://www.presidency.ucsb.edu/).

Following the example of Jerid Francom (https://francojc.github.io/2015/03/01/web-scraping-with-rvest-in-r/), we used Selectorgadget (http://selectorgadget.com/) to choose the links we would like to scrap. For this project, we selected all inaugural addresses of past presidents, nomination speeches of major party candidates and farewell addresses. We also included several public speeches from Donald Trump for our textual analysis of presidential speeches.

The inaugural speeches dates are as follows.

Hide

Hide

```
### Inauguaral speeches
main.page <- read_html(x = "http://www.presidency.ucsb.edu/inaugurals.php")
# Get link URLs
# f.speechlinks is a function for extracting links from the list of speeches.
inaug=f.speechlinks(main.page)
#head(inaug)
as.Date(inaug[,1], format="%B %e, %Y")
```

```
 [1] "1789-04-30" "1793-03-04" "1797-03-04" "1801-03-04" "1805-03-04" "1809-03-04" "1
813-03-04"
 [8] "1817-03-04" "1821-03-04" "1825-03-04" "1829-03-04" "1833-03-04" "1837-03-04" "1
841-03-04"
[15] "1845-03-04" "1849-03-05" "1853-03-04" "1857-03-04" "1861-03-04" "1865-03-04" "1
869-03-04"
[22] "1873-03-04" "1877-03-05" "1881-03-04" "1885-03-04" "1889-03-04" "1893-03-04" "1
897-03-04"
[29] "1901-03-04" "1905-03-04" "1909-03-04" "1913-03-04" "1917-03-04" "1921-03-04" "1
925-03-04"
[36] "1929-03-04" "1933-03-04" "1937-01-20" "1941-01-20" "1945-01-20" "1949-01-20" "1
953-01-20"
[43] "1957-01-21" "1961-01-20" "1965-01-20" "1969-01-20" "1973-01-20" "1977-01-20" "1
981-01-20"
[50] "1985-01-21" "1989-01-20" "1993-01-20" "1997-01-20" "2001-01-20" "2005-01-20" "2
009-01-20"
[57] "2013-01-21" "2017-01-20" NA
```

Hide

Hide

```
inaug=inaug[-nrow(inaug),] # remove the last line, irrelevant due to error.
#### Nomination speeches
main.page=read_html("http://www.presidency.ucsb.edu/nomination.php")
# Get link URLs
nomin <- f.speechlinks(main.page)
nomin <- nomin[-47,]
#head(nomin)
#
#### Farewell speeches
main.page=read_html("http://www.presidency.ucsb.edu/farewell_addresses.php")
# Get link URLs
farewell <- f.speechlinks(main.page)
#head(farewell)
```

Using speech metadata posted on http://www.presidency.ucsb.edu/ (http://www.presidency.ucsb.edu/), we prepared CSV data sets for the speeches we will scrap.

Hide

Hide

```
inaug.list=read.csv("../data/inauglist.csv", stringsAsFactors = FALSE)
nomin.list=read.csv("../data/nominlist.csv", stringsAsFactors = FALSE)
farewell.list=read.csv("../data/farewelllist.csv", stringsAsFactors = FALSE)
```

We assemble all scrapped speeches into one list. Note here that we don’t have the full text yet, only the links to full text transcripts.

Scrap the texts of speeches from the speech URLs.

Hide

Hide

```
speech.list=rbind(inaug.list, nomin.list, farewell.list)
speech.list$type=c(rep("inaug", nrow(inaug.list)),
                   rep("nomin", nrow(nomin.list)),
                   rep("farewell", nrow(farewell.list)))
speech.url=rbind(inaug, nomin, farewell)
speech.list=cbind(speech.list, speech.url)
```

Based on the list of speeches, we scrap the main text part of the transcript's html page. For simple html pages of this kind, Selectorgadget (http://selectorgadget.com/) is very convenient for identifying the html node that `rvest` can use to scrap its content. For reproducibility, we also save our scrapped speeches into our local folder as individual speech files.

Hide

Hide

```
# Loop over each row in speech.list
speech.list$fulltext=NA
for(i in seq(nrow(speech.list))) {
  text <- read_html(speech.list$urls[i]) %>% # load the page
    html_nodes(".displaytext") %>% # isloate the text
    html_text() # get the text
  speech.list$fulltext[i]=text
  # Create the file name
  filename <- paste0("../data/fulltext/",
                     speech.list$type[i],
                     speech.list$File[i], "-",
                     speech.list$Term[i], ".txt")
  sink(file = filename) %>% # open file to write
  cat(text)  # write the file
  sink() # close the file
}
speech.list[which(speech.list$File=="GroverCleveland-II"),]$File <- "GroverCleveland"
speech.list[which(speech.list$File=="GroverCleveland-I"),]$File <- "GroverCleveland"
```

Hide

Hide

```
speech1=paste(readLines("../data/fulltext/SpeechDonaldTrump-NA.txt",
                  n=-1, skipNul=TRUE),
              collapse=" ")
speech2=paste(readLines("../data/fulltext/SpeechDonaldTrump-NA2.txt",
                  n=-1, skipNul=TRUE),
              collapse=" ")
speech3=paste(readLines("../data/fulltext/PressDonaldTrump-NA.txt",
                  n=-1, skipNul=TRUE),
              collapse=" ")
Trump.speeches=data.frame(
  President=rep("Donald J. Trump", 3),
  File=rep("DonaldJTrump", 3),
  Term=rep(0, 3),
  Party=rep("Republican", 3),
  Date=c("August 31, 2016", "September 7, 2016", "January 11, 2017"),
  Words=c(word_count(speech1), word_count(speech2), word_count(speech3)),
  Win=rep("yes", 3),
  type=rep("speeches", 3),
  links=rep(NA, 3),
  urls=rep(NA, 3),
  fulltext=c(speech1, speech2, speech3)
)
speech.list=rbind(speech.list, Trump.speeches)
```

For simpler visualization, we divide presidents into two parts: a subset of the presidents gave inaugural speeches during recessions and others. We put “*" behind the presidents’ name in the first group for simpler visualization.

Hide

Hide

```
sel.comparison=c("BarackObama","RonaldReagan","JohnFKennedy","HarrySTruman",
                 "WilliamMcKinley","AndrewJackson","AbrahamLincoln","JamesKPolk",
                 "GroverCleveland","WilliamHenryHarrison","JohnQuincyAdams","JamesMon
roe",
                 "ThomasJefferson","JohnAdams")
sel.other <- setdiff(unique(sentence.list$File),sel.comparison)
for(i in 1:length(sel.comparison)){
  for(j in 1:length(speech.list$File)){
    if(speech.list$File[j] == sel.comparison[i]){
      speech.list$File[j] <- paste(sel.comparison[i],"*")
    }
  }
}
sel.comparison <- paste(sel.comparison, "*")
```

Data Processing — generate list of sentences

We will use sentences as units of analysis for this project, as sentences are natural languge units for organizing thoughts and ideas.

We assign an sequential id to each sentence in a speech (`sent.id`) and also calculated the number of words in each sentence as *sentence length* (`word.count`).

Hide

Hide

```
sentence.list=NULL
for(i in 1:nrow(speech.list)){
  sentences=sent_detect(speech.list$fulltext[i],
                        endmarks = c("?", ".", "!", "|",";"))
  if(length(sentences)>0){
    emotions=get_nrc_sentiment(sentences)
    word.count=word_count(sentences)
    # colnames(emotions)=paste0("emo.", colnames(emotions))
    # in case the word counts are zeros?
    emotions=diag(1/(word.count+0.01))%*%as.matrix(emotions)
    sentence.list=rbind(sentence.list,
                        cbind(speech.list[i,-ncol(speech.list)],
                              sentences=as.character(sentences),
                              word.count,
                              emotions,
                              sent.id=1:length(sentences)
                              )
    )
  }
}
```

Some non-sentences exist in raw data due to erroneous extra end-of-sentence marks. Remove them.

Step 1: WordCloud

Read in the speeches.

Text processing

Inspect an overall wordcloud


will
government
people
states
can
may
upon
union
must
power
great
united
constitution
time
shall
peace
country
public
one
every
nation
now
world
war
rights
interests
made
nations
national
powers
free
foreign
new
among
general
good
duty
citizens
principles
state
executive
american
america
freedom
liberty
right
spirit
just
present
man
principle
hope
support
system
president
administration
human
governments
without
justice
let
far
commerce
work
whole
political
law
years
future
part
confidence
never
duties
less
men
first
best
congress
make
believe
might
well
within
life
force
party
fellowcitizens
much
revenue
many
long
policy
federal
institutions
laws
equal
ever
high
history
still

These are the wordcloud of presidents’ inaugural speeches of the first group in blue and of the other group in green.The words related to economy, like “revenue”, “interests”, “support”, show up in the first one. So, we can conclude that when there was a recession, presidents did say more words related to economy than the others.

Step 2: Data analysis — Topic modeling

For topic modeling, we prepare a corpus of sentence snipets as follows. For each speech, we start with sentences and prepare a snipet with a given sentence with the flanking sentences.

Hide

Hide

```
corpus.list=sentence.list[2:(nrow(sentence.list)-1), ]
sentence.pre=sentence.list$sentences[1:(nrow(sentence.list)-2)]
sentence.post=sentence.list$sentences[3:(nrow(sentence.list)-1)]
corpus.list$snipets=paste(sentence.pre, corpus.list$sentences, sentence.post, sep=" "
)
rm.rows=(1:nrow(corpus.list))[corpus.list$sent.id==1]
rm.rows=c(rm.rows, rm.rows-1)
corpus.list=corpus.list[-rm.rows, ]
```

** Text mining**

Hide

Hide

```
docs <- Corpus(VectorSource(corpus.list$snipets))
```

Text basic processing Adapted from https://eight2late.wordpress.com/2015/09/29/a-gentle-introduction-to-topic-modeling-using-r/ (https://eight2late.wordpress.com/2015/09/29/a-gentle-introduction-to-topic-modeling-using-r/).

Hide

Hide

```
#remove potentially problematic symbols
docs <-tm_map(docs,content_transformer(tolower))
#remove punctuation
docs <- tm_map(docs, removePunctuation)
#Strip digits
docs <- tm_map(docs, removeNumbers)
#remove stopwords
docs <- tm_map(docs, removeWords, stopwords("english"))
#remove whitespace
docs <- tm_map(docs, stripWhitespace)
#Stem document
docs <- tm_map(docs,stemDocument)
```

Topic modeling

Gengerate document-term matrices.

Hide

Hide

```
dtm <- DocumentTermMatrix(docs)
#convert rownames to filenames#convert rownames to filenames
rownames(dtm) <- paste(corpus.list$type, corpus.list$File,
                       corpus.list$Term, corpus.list$sent.id, sep="_")
rowTotals <- apply(dtm , 1, sum) #Find the sum of words in each Document
dtm  <- dtm[rowTotals> 0, ]
corpus.list=corpus.list[rowTotals>0, ]
```

Run LDA

Hide

Hide

```
#Set parameters for Gibbs sampling
burnin <- 4000
iter <- 2000
thin <- 500
seed <-list(2003,5,63,100001,765)
nstart <- 5
best <- TRUE
#Number of topics
k <- 15
#Run LDA using Gibbs sampling
ldaOut <-LDA(dtm, k, method="Gibbs", control=list(nstart=nstart,
                                                 seed = seed, best=best,
                                                 burnin = burnin, iter = iter,
                                                 thin=thin))
#write out results
#docs to topics
ldaOut.topics <- as.matrix(topics(ldaOut))
table(c(1:k, ldaOut.topics))
```

```
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15 
1691 2143 1761 1953 2049 1162 1262 1073 1298 1188  912 1609 1315  993  572 
```

Hide

Hide

```
write.csv(ldaOut.topics,file=paste("../output/LDAGibbs",k,"DocsToTopics.csv"))
#top 6 terms in each topic
ldaOut.terms <- as.matrix(terms(ldaOut,20))
write.csv(ldaOut.terms,file=paste("../output/LDAGibbs",k,"TopicsToTerms.csv"))
#probabilities associated with each topic assignment
topicProbabilities <- as.data.frame(ldaOut@gamma)
write.csv(topicProbabilities,file=paste("../output/LDAGibbs",k,"TopicProbabilities.cs
v"))
```

Hide

Hide

Hide

```
terms.beta=ldaOut@beta
terms.beta=scale(terms.beta)
topics.terms=NULL
for(i in 1:k){
  topics.terms=rbind(topics.terms, ldaOut@terms[order(terms.beta[i,], decreasing = TR
UE)[1:7]])
}
```

Based on the most popular terms and the most salient terms for each topic, we assign a hashtag to each topic. This part require manual setup as the topics are likely to change.

Hide

Hide

```
topics.hash=c("Economy", "America", "Defense", "Belief", "Election", "Patriotism", "U
nity", "Government", "Reform", "Temporal", "WorkingFamilies", "Freedom", "Equality",
"Misc", "Legislation")
corpus.list$ldatopic=as.vector(ldaOut.topics)
corpus.list$ldahash=topics.hash[ldaOut.topics]
colnames(topicProbabilities)=topics.hash
corpus.list.df=cbind(corpus.list, topicProbabilities)
```

Matrix of topics Choosing the following topics to generate the matrix of topics to see if presidents in the first group cared more about economy than the ones in the other group: “Economy”, “Equality”, “Defense”, “Legislation”, “Government”, “Reform”, “Freedom” and “WorkingFamilies”

Hide

Hide

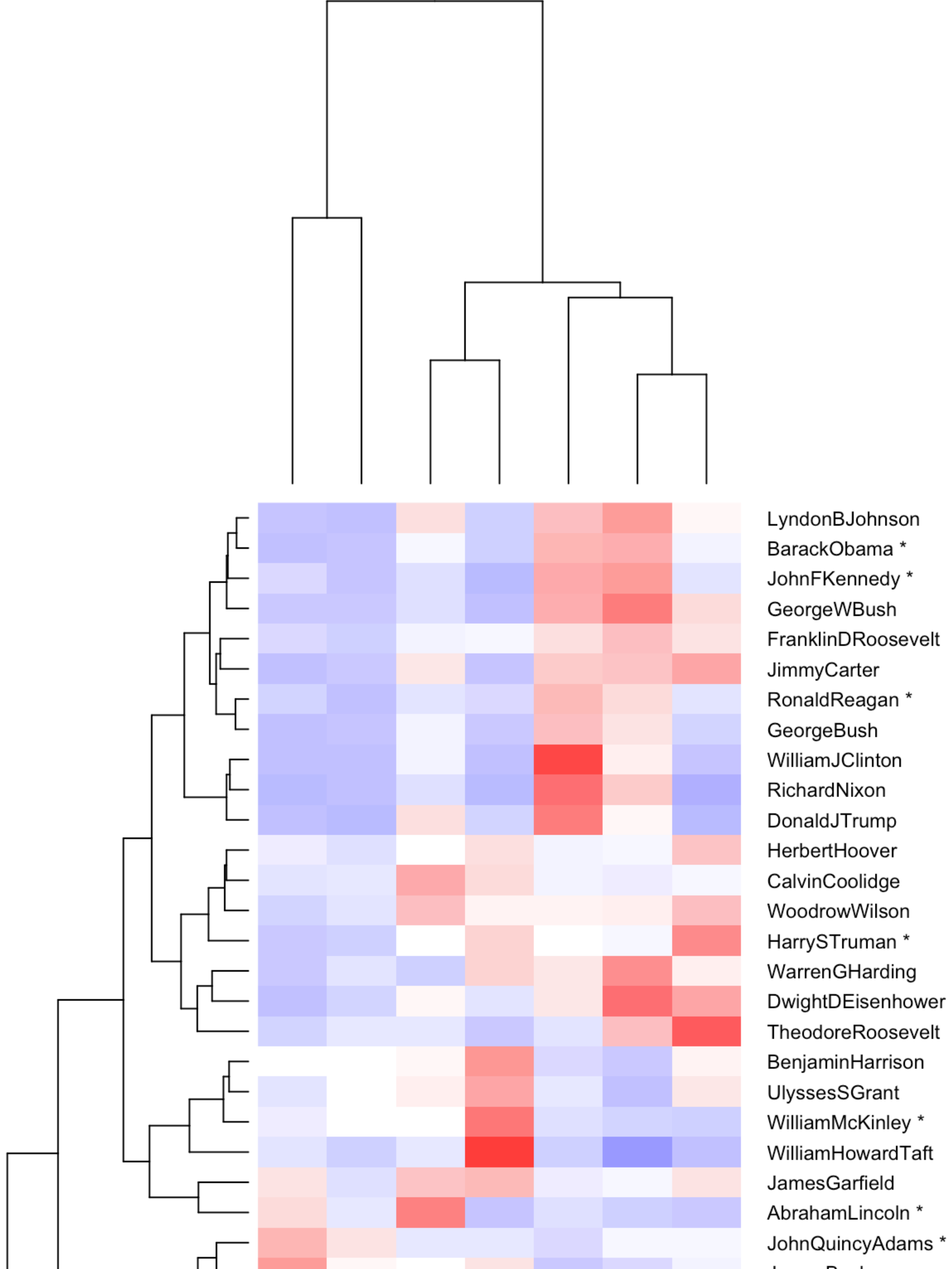
```
topic.plot=c(1, 13, 9, 11, 8, 3, 7)
print(topics.hash[topic.plot])
```

```
[1] "Economy"         "Equality"        "Reform"          "WorkingFamilies" "Governme
nt"
[6] "Defense"         "Unity"
```

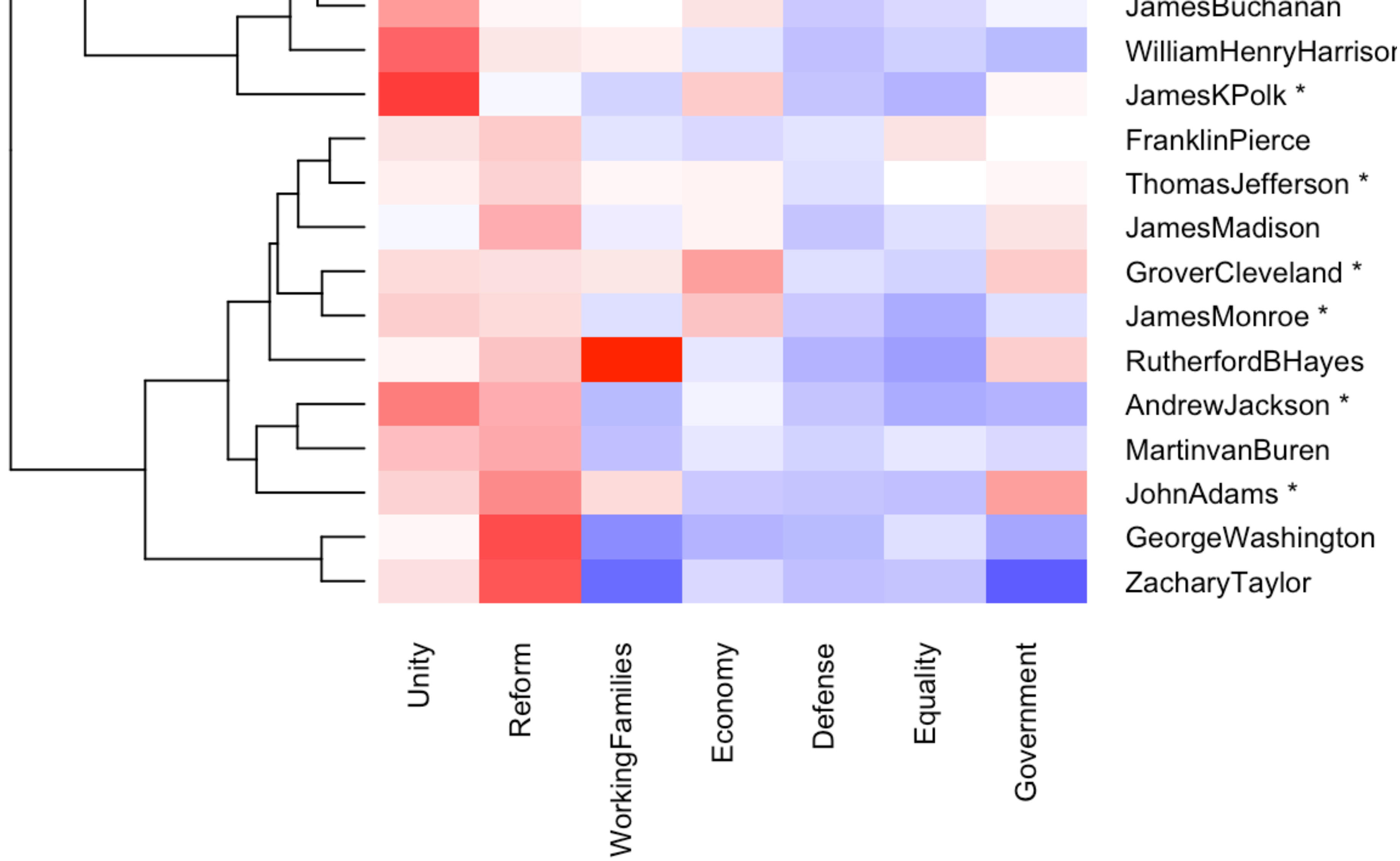
Hide

Hide

```
heatmap.2(as.matrix(topic.summary[,topic.plot+1]),
          scale = "column", key=F,
          col = bluered(100),
          cexRow = 0.9, cexCol = 0.9, margins = c(8, 8),
          trace = "none", density.info = "none")
```



LyndonBJohnson
BarackObama *
JohnFKennedy *
GeorgeWBush
FranklinDRoosevelt
JimmyCarter
RonaldReagan *
GeorgeBush
WilliamJClinton
RichardNixon
DonaldJTrump
HerbertHoover
CalvinCoolidge
WoodrowWilson
HarrySTruman *
WarrenGHarding
DwightDEisenhower
TheodoreRoosevelt
BenjaminHarrison
UlyssesSGrant
WilliamMcKinley *
WilliamHowardTaft
JamesGarfield
AbrahamLincoln *
JohnQuincyAdams *



Because economy as a topic is related closely to working families, and there's an inner connection between these two words, we consider they both as economy related topics. As the heatmap shown above, the colors in economy and working families do seem darker mostly related to the presidents of the first group.

** Clustering of topics**

Hide

Hide

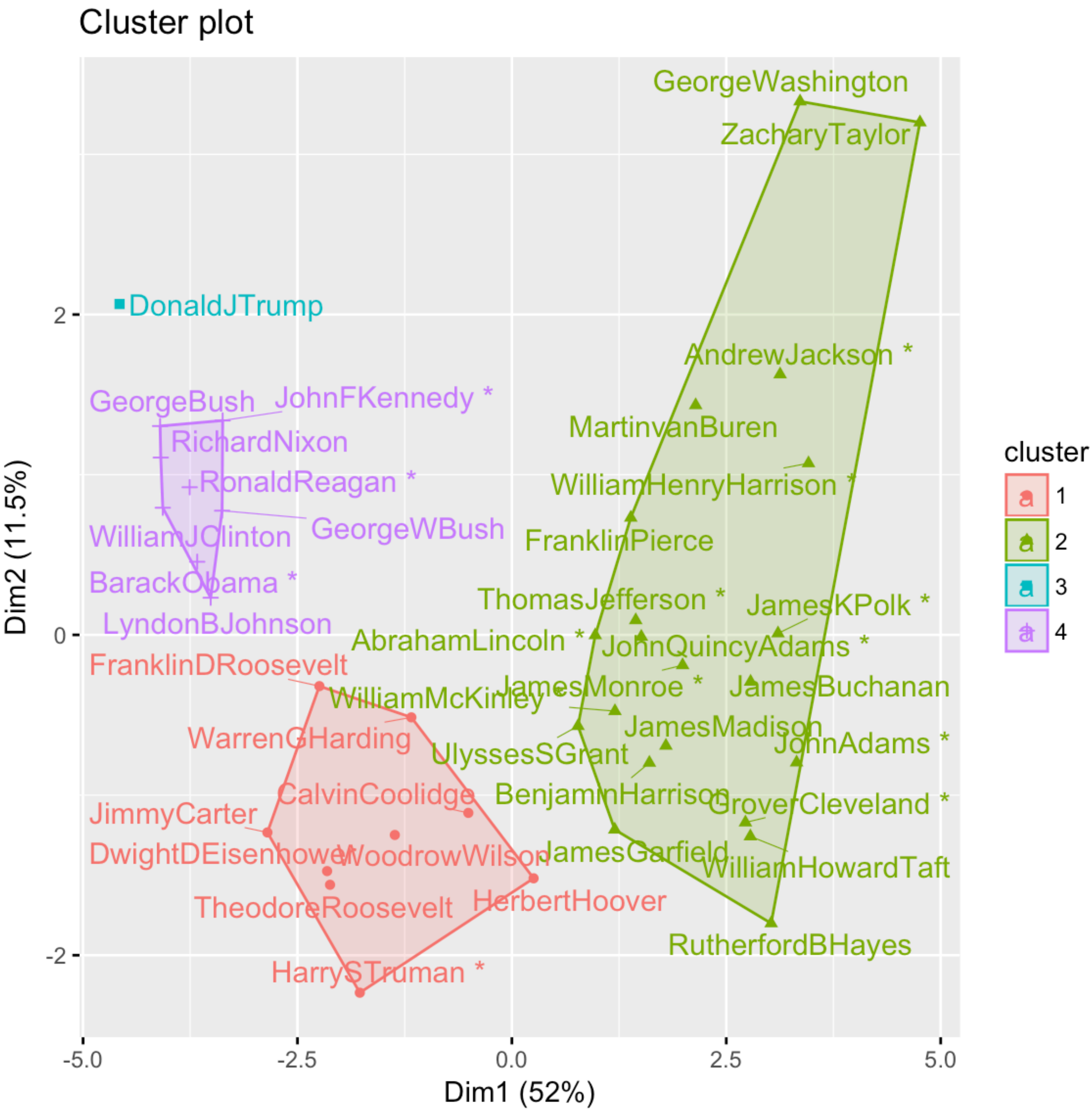
```
presid.summary=tbl_df(corpus.list.df)%>%
  filter(type=="inaug")%>%
  select(File, Economy:Legislation)%>%
  group_by(File)%>%
  summarise_each(funs(mean))
```

```
`summarise_each()` is deprecated.
Use `summarise_all()`, `summarise_at()` or `summarise_if()` instead.
To map `funs` over all variables, use `summarise_all()`
```

Hide

Hide

```
presid.summary=as.data.frame(presid.summary)
rownames(presid.summary)=as.character((presid.summary[,1]))
km.res=kmeans(scale(presid.summary[,-1]), iter.max=200,
              4)
fviz_cluster(km.res,
             stand=T, repel= TRUE,
             data = presid.summary[,-1],
             show.clust.cent=FALSE)
```



Using cluster analysis and dividing presidents into four groups, most ones in the first group stay in the same cluster (green), but this cluster also contains many presidents of the other group. Donald Trump stands out from others. It is interesting, but we can't conclude there's a clear difference between the topics of the two groups.

Step 3: Sentiment Analysis

For each extracted sentence, we apply sentiment analysis using NRC sentiment lexion (http://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm). "The NRC Emotion Lexicon is a list of English words and their associations with eight basic emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) and two sentiments (negative and positive). The annotations were manually done by crowdsourcing."

Hide

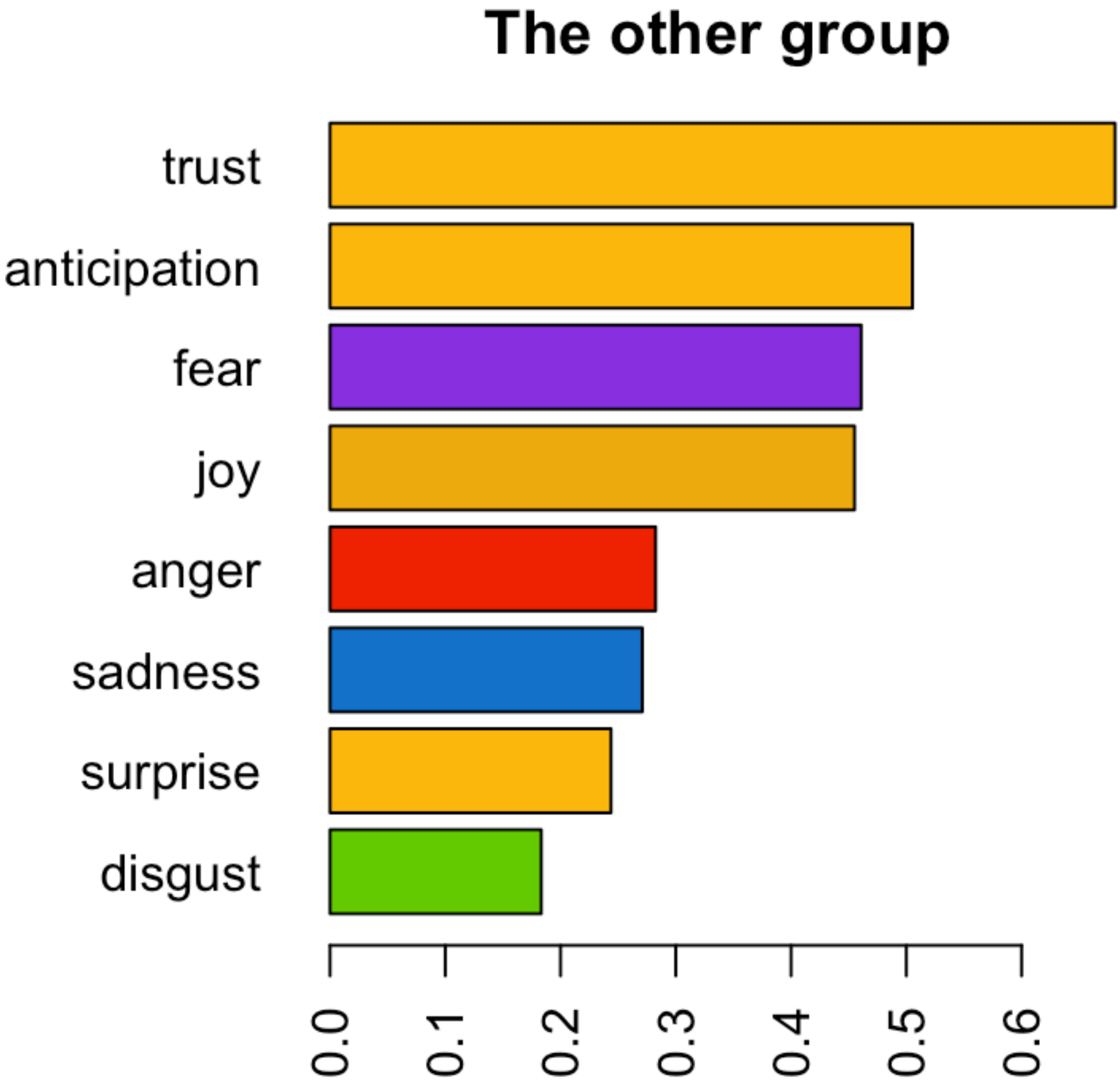
Hide

```
par(mar=c(4, 6, 2, 1))
emo.means=colMeans(select(sentence.list%>%filter(type=="inaug", File%in%sel.other), a
nger:trust)>0.01)
col.use=c("red2", "darkgoldenrod1",
            "chartreuse3", "blueviolet",
            "darkgoldenrod2", "dodgerblue3",
            "darkgoldenrod1", "darkgoldenrod1")
barplot(emo.means[order(emo.means)], las=2, col=col.use[order(emo.means)], horiz=T, m
ain="The first group")
par(mar=c(4, 6, 2, 1))
```

Hide

Hide

```
emo.means=colMeans(select(sentence.list%>%filter(type=="inaug", File%in%sel.compariso
n), anger:trust)>0.01)
col.use=c("red2", "darkgoldenrod1",
            "chartreuse3", "blueviolet",
            "darkgoldenrod2", "dodgerblue3",
            "darkgoldenrod1", "darkgoldenrod1")
barplot(emo.means[order(emo.means)], las=2, col=col.use[order(emo.means)], horiz=T, m
ain="The other group")
```



As shown above, the first group shows more joy and less fear than the other group.

Clustering of sentiment

Hide

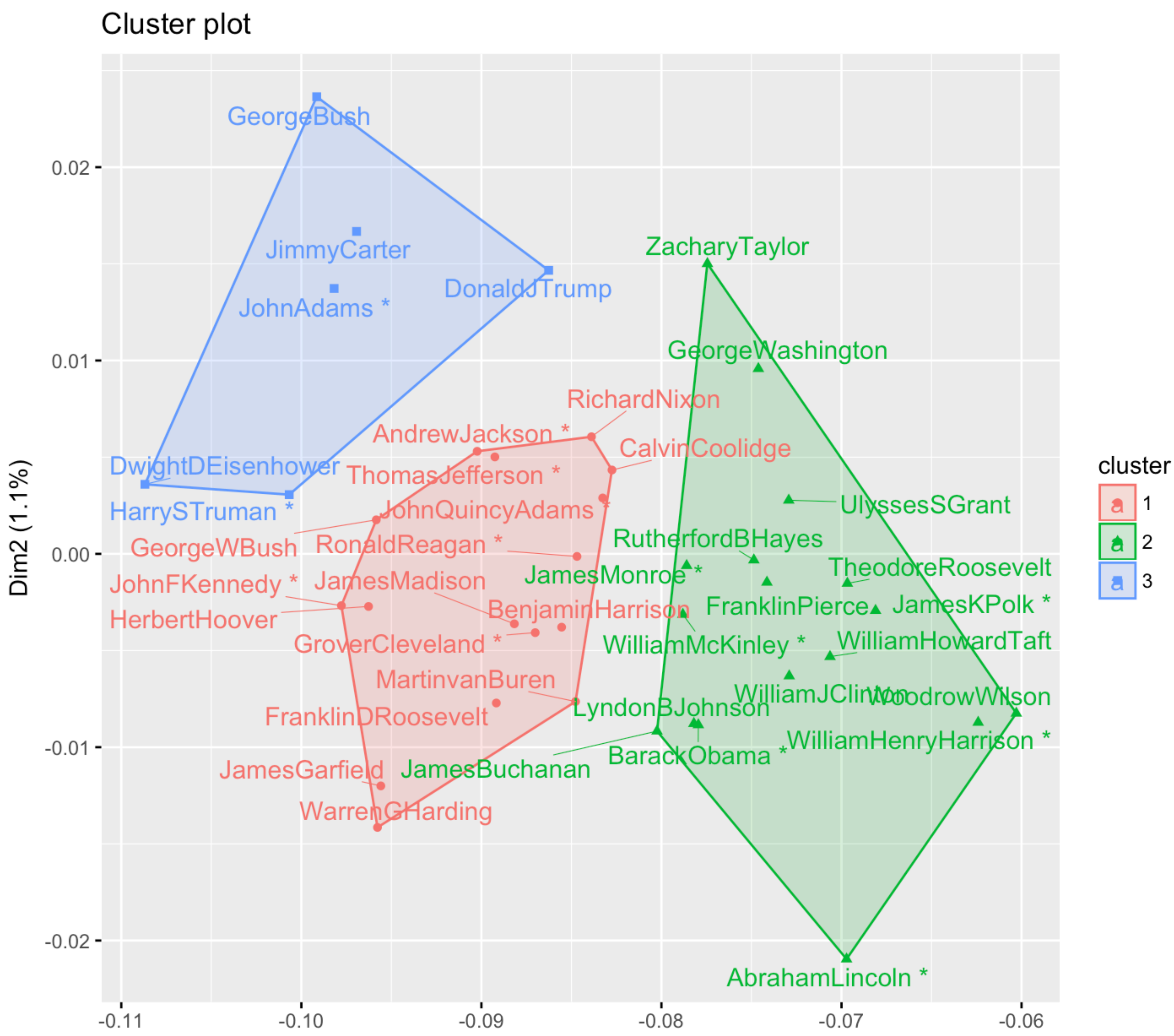
Hide

```
presid.summary=tbl_df(sentence.list)%>%
  filter(type=="inaug")%>%
  #group_by(paste0(type, File))%>%
  group_by(File)%>%
  summarise(
    anger=mean(anger),
    anticipation=mean(anticipation),
    disgust=mean(disgust),
    fear=mean(fear),
    joy=mean(joy),
    sadness=mean(sadness),
    surprise=mean(surprise),
    trust=mean(trust)
    #negative=mean(negative),
    #positive=mean(positive)
  )
presid.summary=as.data.frame(presid.summary)
rownames(presid.summary)=as.character((presid.summary[,1]))
km.res=kmeans(presid.summary[,-1], iter.max=200,
              2)
fviz_cluster(km.res,
             stand=F, repel= TRUE,
             data = presid.summary[,-1], xlab="", xaxt="n",
             show.clust.cent=FALSE)
```

Hide

Hide

```
presid.summary=as.data.frame(presid.summary)
rownames(presid.summary)=as.character((presid.summary[,1]))
km.res=kmeans(presid.summary[,-1], iter.max=200,
               3)
fviz_cluster(km.res,
              stand=F, repel= TRUE,
              data = presid.summary[,-1], xlab="", xaxt="n",
              show.clust.cent=FALSE)
```



Using cluster analysis and dividing presidents into two and three clusters, there's not obvious difference between the two goups I want to analyze, as the first group is evenly distributed.

Summary of analyzing the presidents' inaugural speeches

1.According to the Word Cloud analysis, during economic recession days, presidents used more economy related words. It is no wonder that in such hard time, the elected presidents would show their care about economy to comfort their people.

2.By using LDA model to get 7 topics from presidents' inaugural, not all presidents emphasized economic topics during recessions. Although most of them talked more economic topics than the presidents who did not take office during recessions, the clustering analysis shows there's not obvious differences between these two groups, and only Donald Trump chose really different topic distribution from others. Perhaps besides the economy, many other things matter.

3.As for sentiment analysis, presidents’ inaugural speeches sounded more joyful when the economic background of United States were not optimistic. This is reasonable, as the presidents were supposed to comfort people and gave them confidence about future.