Lesson 11 - Latent Dirichlet Allocation (LDA)

Dr. Jeffrey Strickland

9/12/2018

PART I: BLOG TOPIC ANALYSIS USING LDA

Topic modeling focuses on the problem of classifying sets of documents into themes, either manually or automatically (preferred). It is a way of identifying patterns in a corpus-technically, this is text mining. We take our corpus and group words across the corpus into 'topics,' using a text mining algorithm or tool.

In natural language processing, latent Dirichlet allocation (LDA) is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. A topic model which discovers underlying topics in a collection of documents and infers word probabilities in topics

We will use a collection of 20 posts from my LinkedIn blog as an example corpus, call "blog_corpus." The corpus can be downloaded at https://github.com/stricje1/VIT_University/tree/master/Data_Analytics_2018/data.

```
#load text mining library
library(tm)

## Loading required package: NLP
```

Set working Directory

see your working dierct(modify path as needed)

```
#setwd("C:/Users/jeff/Documents/VIT_Course_Material/Data_Analytics_2018/code/blog_corpus/")
```

Here, we get a listing of .txt files in directory.

```
filenames <- list.files(getwd(),pattern="*.txt")
```

Read files into a character vector

```
files <- lapply(filenames,readLines)
```

create corpus from vector

```
docs <- Corpus(VectorSource(files))
```

#inspect a particular document in corpus

```
writeLines(as.character(docs[1]))
```

```
## c("10 More Signs that you might be a Data Scientist", "Why are people so
down on Data Scientists? Probably because they don't know what one looks
like. There are a lot of folks out there calling themselves Data Scientist
just because they had some introductory statistic courses, and maybe they can
make really cool charts in Excel. All that makes you is someone who had some
introductory statistic courses and can make some really cool charts in Excel.
SO what might a real Data Scientist look like? Here are some more signs that
you might \"really\" be a Data Scientist (see 10 Signs that you might be a
Data Scientist, October 7, 2014).",
## "1.\tYou have a title of Operation Research Analyst, Principal Business
Analyst, Senior Risk Analyst, or Statistician, Data Architect, Senior Data
Analyst, and so on.", "2.\tYou have a graduate degree or graduate course work
in a related field, i.e., Analytics, Predictive Modeling, Risk Analysis,
Applied Statistics, Database Architecture, etc.", "3.\tYou have a
professionally taken greyscale photo on your LinkedIn profile. Who does
that?", "4.\tIn addition to Data Science, you include some or all or the
following skills on your profile: SPSS Modeler, SAS Programming, Regression
Analysis, Statistical Modeling, Logistics Regression, Machine Learning, SQL,
Python, etc.",
## "5.\tYour boss or customer listens to you and takes action when you
present your analysis and recommendations.", "6.\tNot only can you crunch
numbers with the best of them, you can logically present the results of your
analysis in plain language, or at least your customerâ\200\231s language.",
"7.\tYou write incredibly Geek-like post on LinkedIn, whether there are many
viewers or not. Who does that?", "8.\tProfit was made or money was saved (or
for non-profits some measure of performance was achieved) based on your
work.",
## "9.\tYou belong to a LinkedIn group for predictive modeling, prescriptive
modeling, analytics, business analytics, etc.", "10.\tOne of your hobbies is
data analysis and modeling. Who would do that?")
## list(language = "en")
## list()
```

Start Preprocessing

The next preprocessing steps include converting to lower case, removing special characters/symbols, removing punctuation, stripping digits, removing stop words, abd removing white spaces.

First, we transform the corpus to lower case.

```
docs <-tm_map(docs,content_transformer(tolower))
```

Second, we remove punctuation.

```
docs <- tm_map(docs, removePunctuation)
```

Third, we strip digits from the corpus.

```
docs <- tm_map(docs, removeNumbers)
```

Fourth, we remove stopwords.

```
docs <- tm_map(docs, removeWords, stopwords("english"))
```

Finally, we remove whitespaces.

```
docs <- tm_map(docs, stripWhitespace)
```

It is good practice to check the document preprocessing every now and then.

```
writeLines(as.character(docs[2]))
```

```
## cten signs might data scientist nailed really good definition data
scientist argue need entity poorly defined perform one following task might
data scientist tyou build databased models predictive descriptive
prescriptive tyou perform data mining analysis tyou use data perform
analytics business otherwise tyou can spell data scientist using binary
numbers tyou regularly use sas enterprise guide sas enterprise miner spss
modeler spss statistics tyou can program r use perform data analysis modeling
tyou really hardcore use matlab octave data analysis tyou can spell data
scientist latin ready little sas book tyou entire sas library ipad iphone etc
tyou use data provide added value company customer ten signs might data
scientist tyou title operation research analyst principal business analyst
senior risk analyst statistician tyou graduate degree graduate course work
## list(language = "en")
## list()
```

Stem document

(fix up 1) differences between us and aussie english 2) general errors

```
docs <- tm_map(docs,stemDocument)

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "organiz", replacement = "organ")

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "organis", replacement = "organ")

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "andgovern", replacement = "govern")

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "inenterpris", replacement = "enterpris")

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "team-", replacement = "team")

docs <- tm_map(docs, content_transformer(gsub),
               pattern = "henry", replacement = "henry ")

#define and eliminate all custom stopwords
myStopwords <- c("can", "say","one","way","use",
```

```
                  "also","howev","tell","will",
                  "much","need","take","tend","even",
                  "like","particular","rather","said",
                  "get","well","make","ask","come","end",
                  "first","two","help","often","may",
                  "might","see","someth","thing","point",
                  "post","look","right","now","think",
                  "anoth","put","set","new","good",
                  "want","sure","kind","larg","yes,","day","etc",
                  "quit","sinc","attempt","lack","seen","awar",
                  "littl","ever","moreov","though","found","abl",
                  "enough","far","earli","away","achiev","draw",
                  "last","never","brief","bit","entir","brief",
                  "great","lot")
docs <- tm_map(docs, removeWords, myStopwords)
```

At this point, it is a good idea to inspect a document as a check.

```
writeLines(as.character(docs[3]))
```

```
## c type analyt user sell thursday afternoon analyt user meet becom standard
mayb youâr familiar similar meet key metric everyon follow present present
deck everi number slice dice bar chart pie chart dozen page  differ version
process  whatâ realli happen peopl  question most peopl just took  meet peopl
thank present taken  time  number togeth hung accomplish  declar meet wast
time theyâr wast time analyt mean differ  differ peopl â“ differ analyt user
donât understand peopl consum analyt wonât absorb  lesson critic analyt solut
provid youâr sell differ user mean solut  adapt differ user messag  adapt
analyt user user analyt break  dimens purpos analyt versus skill  user    axe
form box diagram across bottom userâ purpos analyt either exploratori
decisionmak along side userâ technic skill level lower technic skill higher
technic skill lower technic skill user  someon understand metric graph chart
produc advanc analyt output higher technic skill user   understand general
advanc statist output twodimens breakdown creat four major type analyt user
analyst scientist expert execut type analyt user break categori type analyt
user differ  type softwar   differ type support requir  differ analyst letâ
start explor analyst user analyt user statist savvi someon understand
databas queri simpli possess sophist advanc technic skill  statist model
construct role primarili queri data number differ  understand relationship
## list(language = "en")
## list()
```

Create document-term matrix

This code chunk generates the document-term matrix for our corpus, and the next several code chucks set up the document-term matrix for our analysis.

```
dtm <- DocumentTermMatrix(docs)
```

This code chunk converts rownames to filenames

```
rownames(dtm) <- filenames
```

This code chunk collapses matrix by summing over columns

```
freq <- colSums(as.matrix(dtm))
```

This code chunk determines the length, which should be total number of terms.

```
length(freq)

## [1] 2833
```

This code chunk creates the sort order (descending by frequency) for our terms and writes it to our disk.

```
ord <- order(freq,decreasing=TRUE)
write.csv(freq[ord],"word_freq.csv")
```

Load Topic Models Library

Here we load the R-package "topicmodels" that we will be using in our analysis.

```
library(topicmodels)
```

Set Parameters for Gibbs Sampling

In this code chunk we set the parameters for the LDA using Gibbs sampling that we will implement below. This is just a matter of convienance.

```
burnin <- 4000
iter <- 2000
thin <- 500
seed <-list(2003,5,63,100001,765)
nstart <- 5
best <- TRUE
```

Number of Topics

Now we set a rough guess for the number of topics as five. Later, we will use an user function to make a better guess at the optimal number of topics, k.

```
k <- 5
```

Run LDA using Gibbs sampling

Using the parameter we set above, we now implment the LDA model using Gibbs sampling.

```
library(topicmodels)
control=list(nstart=5, seed = list(2003,5,63,100001,765), best=TRUE, burnin =
4000, iter = 2000, thin=500)
ldaOut <-LDA(dtm,5, method="Gibbs", control=control)
```

Write out Results

Now we write the documents to topics:

```
ldaOut.topics <- as.matrix(topics(ldaOut))
write.csv(ldaOut.topics,file=paste("LDAGibbs",k,"DocsToTopics.csv"))
```

Top 6 Terms per Topic

The next code chunk provides the top six terms for each topic.

```
ldaOut.terms <- as.matrix(terms(ldaOut,6))
write.csv(ldaOut.terms,file=paste("LDAGibbs",k,"TopicsToTerms.csv"))
```

Topic Probabilities

Next, we calculate the probabilities associated with each topic assignment.

```
topicProbabilities <- as.data.frame(ldaOut@gamma)
write.csv(topicProbabilities,file=paste("LDAGibbs",k,"TopicProbabilities.csv"
))
```

#Find relative importance of top 2 topics

```
topic1ToTopic2 <- lapply(1:nrow(dtm),function(x)
sort(topicProbabilities[x,])[k]/sort(topicProbabilities[x,])[k-1])
```

#Find relative importance of second and third most important topics

```
topic2ToTopic3 <- lapply(1:nrow(dtm),function(x)
sort(topicProbabilities[x,])[k-1]/sort(topicProbabilities[x,])[k-2])
```

#write to file

```
write.csv(topic1ToTopic2,file=paste("LDAGibbs",k,"Topic1ToTopic2.csv"))
write.csv(topic2ToTopic3,file=paste("LDAGibbs",k,"Topic2ToTopic3.csv"))
```

PART II: 2012 USA PRESIDENTIAL DEBATE USING LDA

[source files available on GitHub]

PRELIMINARIES

Here, we install and load libraries needed for data processing and plotting:

```
if (!require("pacman")) install.packages("pacman")
pacman::p_load_gh("trinker/gofastr")
pacman::p_load(tm, topicmodels, dplyr, tidyr, igraph, devtools, LDAvis,
ggplot2, sentimentr)
```

```
library(scales)
library(Rcpp)
library(tm)
library(topicmodels)
library(dplyr)
library(tidyr)
library(igraph)
library(devtools)
library(LDAvis)
library(ggplot2)
library(sentimentr)
```

Get External Scripts

Next, we call external script containing my own optimal_k functions definitions, using the source function call.

```
source("C:/Users/jeff/Documents/VIT_Course_Material/Data_Analytics_2018/code/optimal_k.R")
```

The content of this external file is included in the Appendix at the end of this document.

Load the Data

Now, we load the data.

```
data(presidential_debates_2012)
head(presidential_debates_2012,5)
```

```
## # A tibble: 5 x 5
##   person tot   time   role    dialogue
##   <fct>  <chr> <fct>  <fct>   <chr>
## 1 LEHRER 1.1   time 1 modera~ We'll talk about specifically about health ~
## 2 LEHRER 1.2   time 1 modera~ But what do you support the voucher system,~
## 3 ROMNEY 2.1   time 1 candid~ What I support is no change for current ret~
## 4 ROMNEY 2.2   time 1 candid~ And the president supports taking dollar se~
## 5 LEHRER 3.1   time 1 modera~ And what about the vouchers?
```

In this code chunk we generate user-defined stopwords that are relevant to our case.

```
stops <- c(
        tm::stopwords("english"),
        tm::stopwords("SMART"),
        "governor", "president", "mister", "obama","romney"
    ) %>%
    gofastr::prep_stopwords()
```

The next code chunk creates the document-term matrix.

```
doc_term_mat <- presidential_debates_2012 %>%
    with(gofastr::q_dtm_stem(dialogue, paste(person, time, sep = "_"))) %>%
```

```
    gofastr::remove_stopwords(stops, stem=TRUE) %>%
    gofastr::filter_tf_idf()
    #gofastr::filter_documenheads()
```

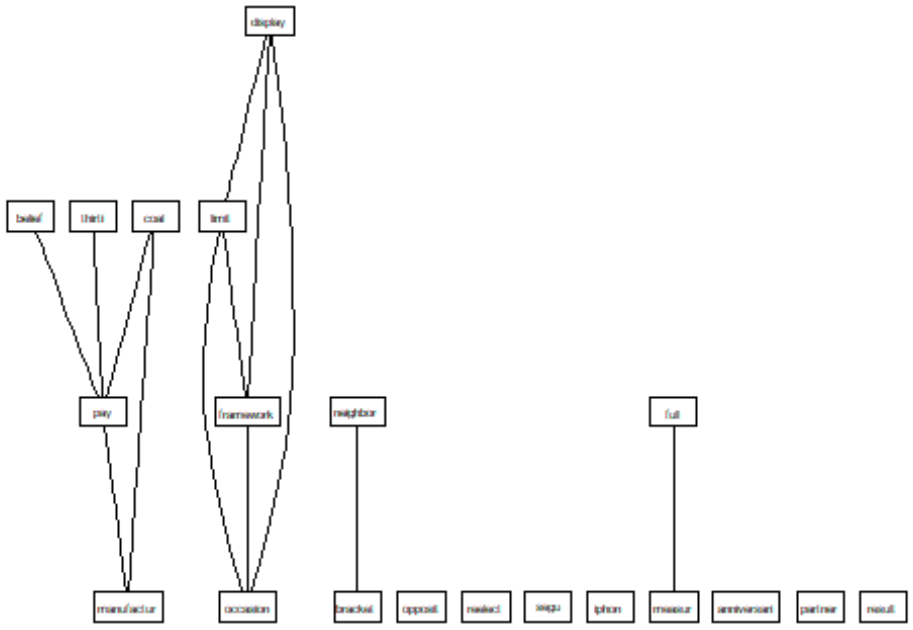
This code chunk established the control list we will use to find optimum_k.

```
control <- list(burnin = 500, iter = 1000, keep = 100)
```

Determine Optimal Number of Topics

Using our external optimal_k function code, we determine the optimal number of topics, which is a little btter than a random guess.

```
k <- optimal_k(doc_term_mat, 40, control = control)

## Grab a cup of coffee this could take a while...

## 10 of 40 iterations (Current: 05:32:19; Elapsed: .1 mins)
## 20 of 40 iterations (Current: 05:32:30; Elapsed: .3 mins; Remaining: ~.7
mins)
## 30 of 40 iterations (Current: 05:32:47; Elapsed: .5 mins; Remaining: ~.4
mins)
## 40 of 40 iterations (Current: 05:33:12; Elapsed: 1 mins; Remaining: ~0
mins)
## Optimal number of topics = 14

print.optimal_k(doc_term_mat, 40, control = control)
```



```
## [1] "A graph with 20 nodes."
```

Harmonic Mean of Log Likelihood

Number of Topics

Number of Topics

Run the Model

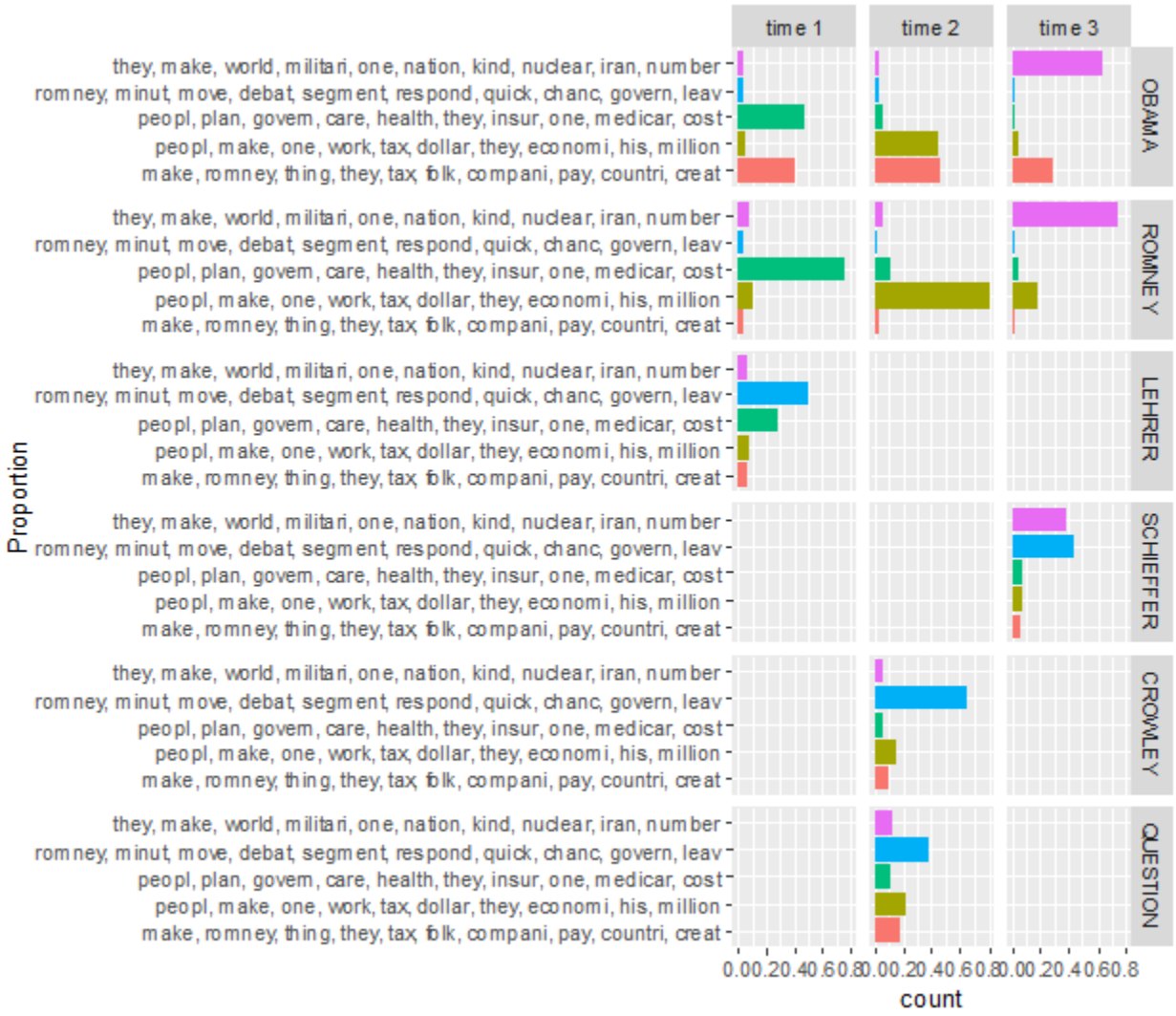
Now, we implement he LDA model using Gibbs sampling and the parameters we have already established.

```
control[["seed"]] <- 100
lda_model <- topicmodels::LDA(doc_term_mat, k=as.numeric(k), method =
"Gibbs",
    control = control)
```

Plot the Topics Per Person & Time

This code chunk is used to generate a plot of the topics per person and time.

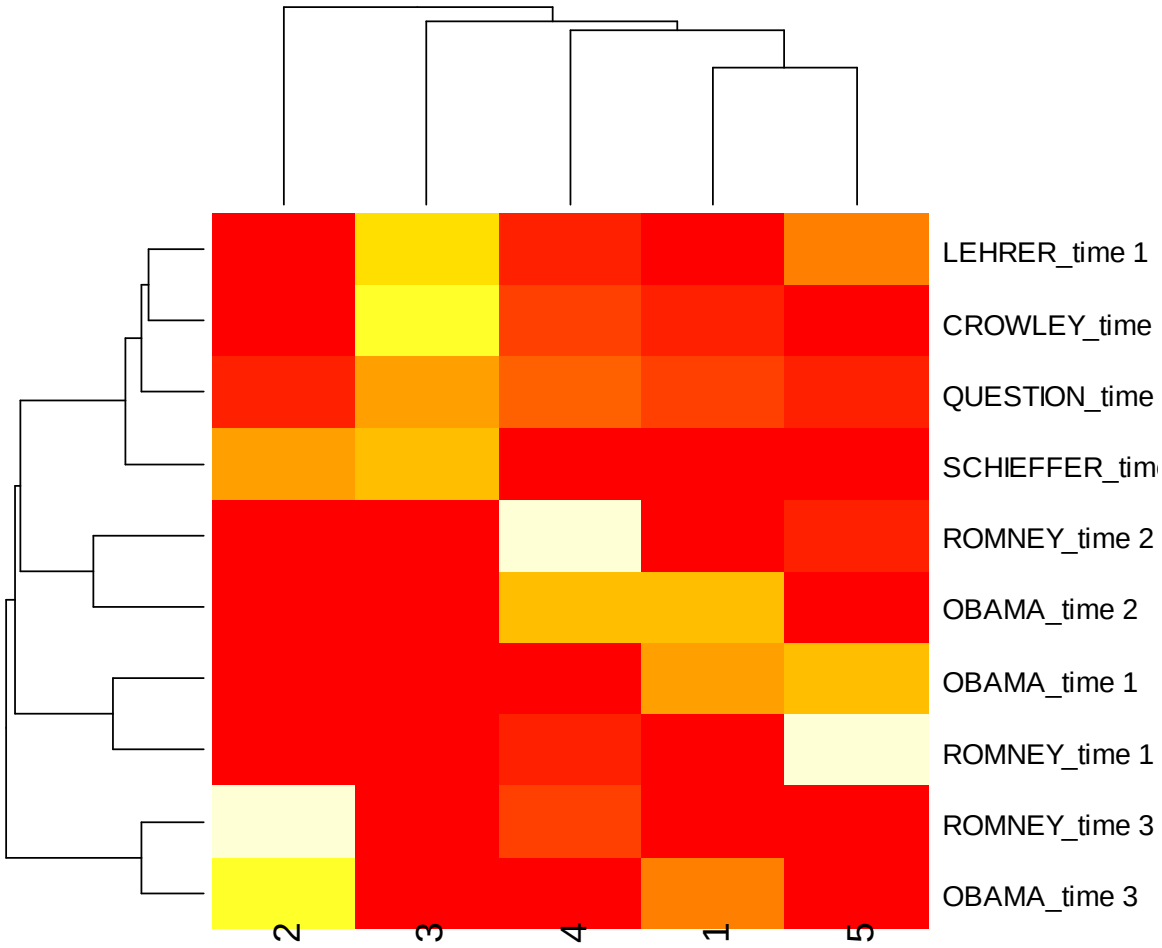
```
topics <- topicmodels::posterior(lda_model, doc_term_mat)[["topics"]]
topic_dat <- dplyr::add_rownames(as.data.frame(topics), "Person_Time")

## Warning: Deprecated, use tibble::rownames_to_column() instead.

colnames(topic_dat)[-1] <- apply(terms(lda_model, 10), 2, paste, collapse =
", ")

tidyr::gather(topic_dat, Topic, Proportion, -c(Person_Time)) %>%
    tidyr::separate(Person_Time, c("Person", "Time"), sep = "_") %>%
    dplyr::mutate(Person = factor(Person,
        levels = c("OBAMA", "ROMNEY", "LEHRER", "SCHIEFFER", "CROWLEY",
"QUESTION" ))
    ) %>%
    ggplot2::ggplot(ggplot2::aes(weight=Proportion, x=Topic, fill=Topic)) +
        ggplot2::geom_bar() +
        ggplot2::coord_flip() +
        ggplot2::facet_grid(Person~Time) +
        ggplot2::guides(fill=FALSE) +
        ggplot2::xlab("Proportion")
```



Plot the Topics Matrix as a Heatmap

Now, we make a heatmap of our results.

```
heatmap(topics, scale = "none")
```



Heatmap of Topics

Topics by Candidate

Topics by Candidate

Network of the Word Distributions Over Topics

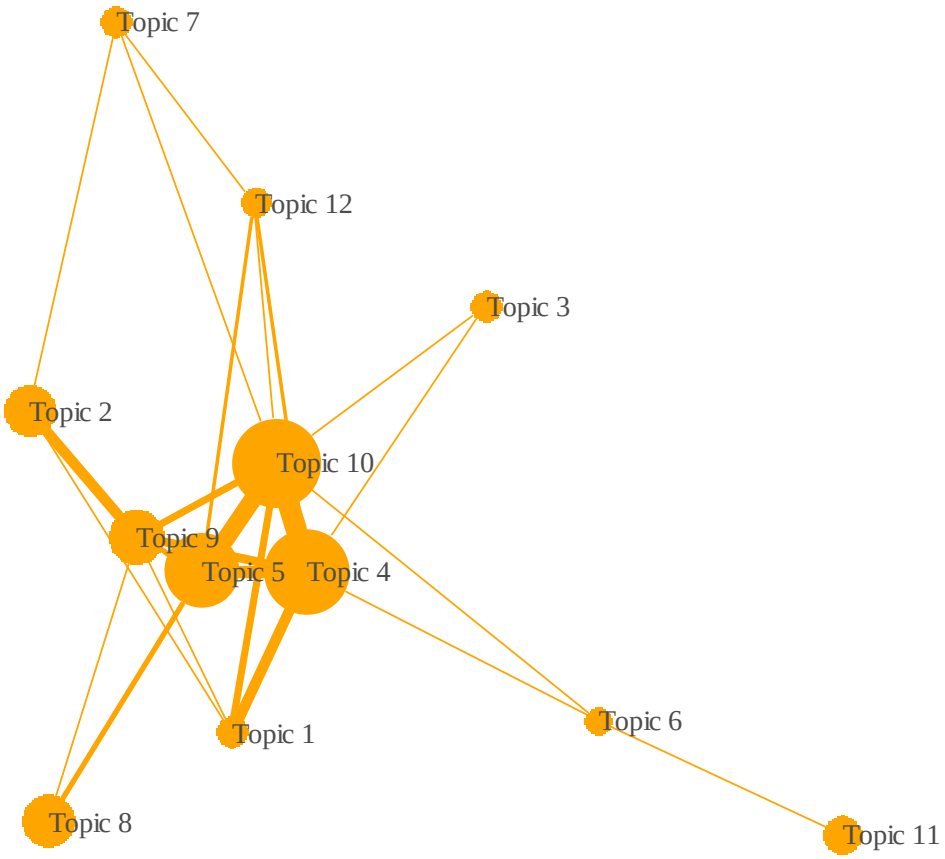
Next, we generate a network of word distributions over topics.

```
post <- topicmodels::posterior(lda_model)
cor_mat <- cor(t(post[["terms"]]))
cor_mat[ cor_mat < .05 ] <- 0
diag(cor_mat) <- 0
```

Strength Between Topics Based On Word Probabilities

```
graph <- graph.adjacency(cor_mat, weighted=TRUE, mode="lower")
graph <- delete.edges(graph, E(graph)[ weight < 0.05])
```

```r
E(graph)$edge.width <- E(graph)$weight*20
V(graph)$label <- paste("Topic", V(graph))
V(graph)$size <- colSums(post[["topics"]]) * 15

par(mar=c(0, 0, 3, 0))
set.seed(110)
plot.igraph(graph, edge.width = E(graph)$edge.width,
    edge.color = "orange", vertex.color = "orange",
    vertex.frame.color = NA, vertex.label.color = "grey30")
title("Strength Between Topics Based On Word Probabilities", cex.main=.8)
```



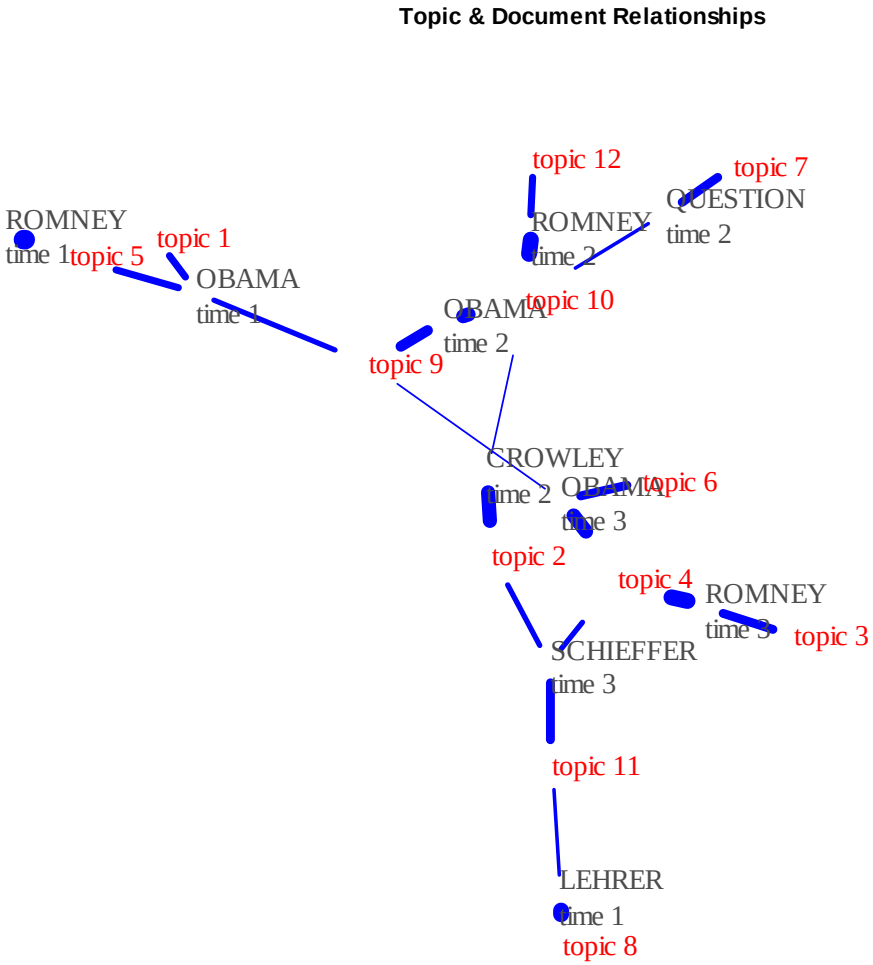
Word Distributions over Topics

Topics by Candidate

Topics by Candidate

This code chunk generates a network of the topics over documents

```
  minval <- .1
  topic_mat <- topicmodels::posterior(lda_model)[["topics"]]

  graph <- graph_from_incidence_matrix(topic_mat, weighted=TRUE)
  graph <- delete.edges(graph, E(graph)[ weight < minval])

  E(graph)$edge.width <- E(graph)$weight*17
  E(graph)$color <- "blue"
  V(graph)$color <- ifelse(grepl("^\\d+$", V(graph)$name), "grey75",
"orange")
  V(graph)$frame.color <- NA
  V(graph)$label <- ifelse(grepl("^\\d+$", V(graph)$name), paste("topic",
V(graph)$name), gsub("_", "\n",   V(graph)$name))
  V(graph)$size <- c(rep(10, nrow(topic_mat)), colSums(topic_mat) * 20)
  V(graph)$label.color <- ifelse(grepl("^\\d+$", V(graph)$name), "red",
"grey30")

  par(mar=c(0, 0, 3, 0))
  set.seed(365)
  plot.igraph(graph, edge.width = E(graph)$edge.width,
      vertex.color = adjustcolor(V(graph)$color, alpha.f = .4))
  title("Topic & Document Relationships", cex.main=.8)
```



Word Topics over Documents

Topics by Candidate

Topics by Candidate

LDAvis of Model

```
source("C:/Users/jeff/Documents/VIT_Course_Material/Data_Analytics_2018/code/
topicmodels2LDAvis.R")
lda_model %>%
    topicmodels2LDAvis() %>%
    LDAvis::serVis()

## Loading required namespace: servr
```

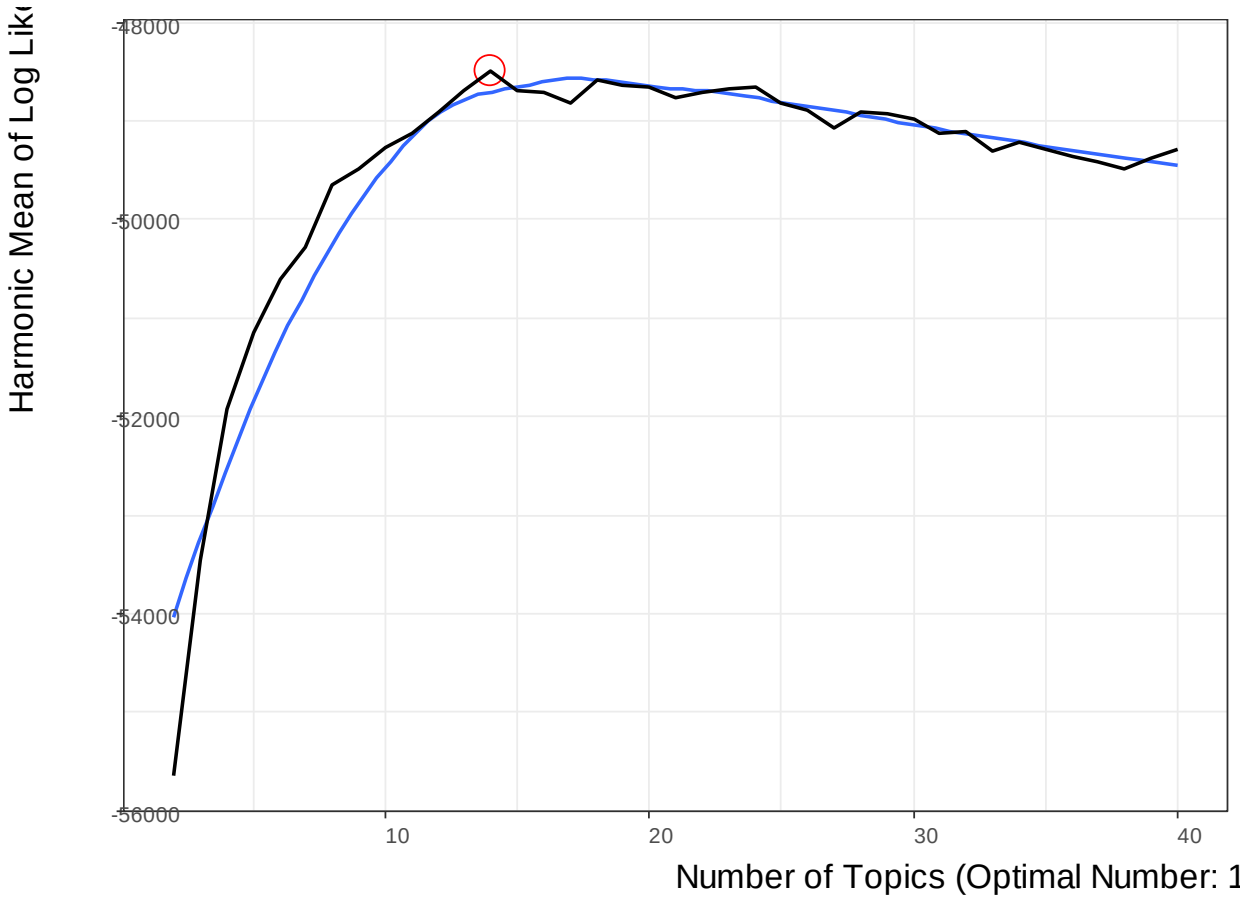
Fitting New Data

```
library(gofastr)
## Create the DocumentTermMatrix for New Data
```

```
doc_term_mat2 <- partial_republican_debates_2015 %>%
    with(gofastr::q_dtm_stem(dialogue, paste(person, location, sep = "_")))
%>%
    gofastr::remove_stopwords(stops, stem=TRUE) %>%
    gofastr::filter_tf_idf() %>%
    gofastr::filter_documents()
```

Run the Model for New Data

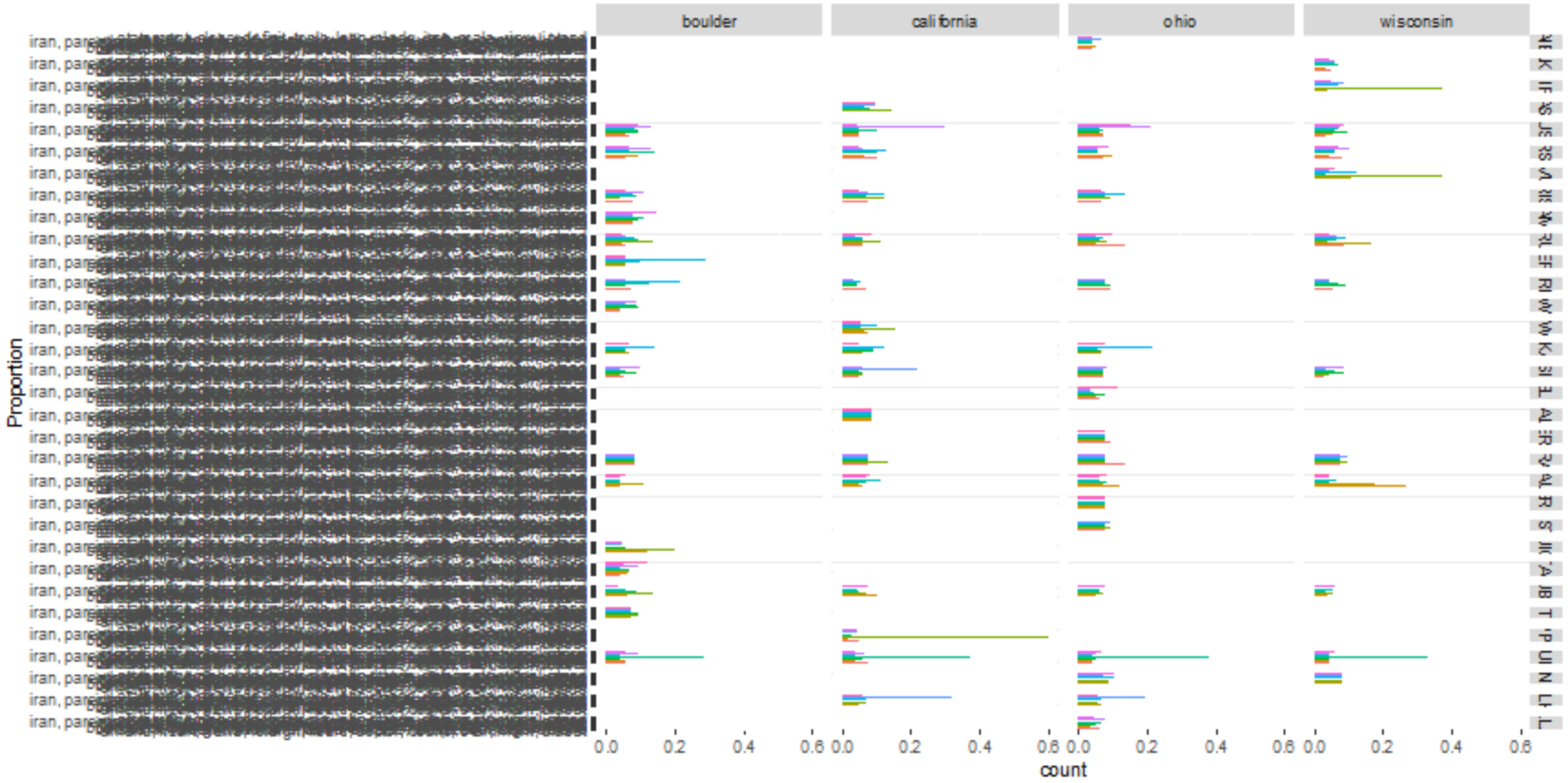
```
control = list(nstart=5, seed = list(2003,5,63,100001,765), best=TRUE, burnin
= 4000, iter = 2000, thin=500)
control2  = list(burnin = 500, iter = 1000, keep = 100)
#control2[["estimate.beta"]] <- FALSE
k <- optimal_k1(doc_term_mat2,40,control=control2)

##
## Grab a cup of coffee this could take a while...

## 10 of 40 iterations (Current: 05:33:21; Elapsed: .1 mins)
## 20 of 40 iterations (Current: 05:33:32; Elapsed: .3 mins; Remaining: ~.7
mins)
## 30 of 40 iterations (Current: 05:33:50; Elapsed: .6 mins; Remaining: ~.4
mins)
## 40 of 40 iterations (Current: 05:34:15; Elapsed: 1 mins; Remaining: ~0
mins)
## Optimal number of topics = 16

plot.optimal_k1(optimal_k1(doc_term_mat2,40,control=control2))

##
## Grab a cup of coffee this could take a while...

## 10 of 40 iterations (Current: 05:34:23; Elapsed: .1 mins)
## 20 of 40 iterations (Current: 05:34:34; Elapsed: .3 mins; Remaining: ~.7
mins)
## 30 of 40 iterations (Current: 05:34:52; Elapsed: .6 mins; Remaining: ~.4
mins)
## 40 of 40 iterations (Current: 05:35:18; Elapsed: 1 mins; Remaining: ~0
mins)
## Optimal number of topics = 23
```




```
lda_model2 <- topicmodels::LDA(doc_term_mat2, k = as.numeric(k), model =
lda_model,
    control = control2)
```

Plot the Topics Per Person & Location for New Data

```
topics2 <- topicmodels::posterior(lda_model2, doc_term_mat2)[["topics"]]
topic_dat2 <- dplyr::add_rownames(as.data.frame(topics2), "Person_Location")

## Warning: Deprecated, use tibble::rownames_to_column() instead.

colnames(topic_dat2)[-1] <- apply(terms(lda_model2, 10), 2, paste, collapse =
", ")

tidyr::gather(topic_dat2, Topic, Proportion, -c(Person_Location)) %>%
    tidyr::separate(Person_Location, c("Person", "Location"), sep = "_") %>%
    ggplot2::ggplot(ggplot2::aes(weight=Proportion, x=Topic, fill=Topic)) +
        ggplot2::geom_bar() +
        ggplot2::coord_flip() +
        ggplot2::facet_grid(Person~Location) +
        ggplot2::guides(fill=FALSE) +
        ggplot2::xlab("Proportion")
```


boulder
california
ohio
wisconsin
Proportion
count
0.0
0.2
0.4
0.6