FIT3152 Data analytics. Tutorial 11: Text analytics

Contents:

- Text analytics, text processing and clustering in R

1. Work through the examples in the lecture slides.

2. Text pre-processing

   Create a Term-Document Matrix of the following data following the process outlined in the lecture notes: (Tokenize, Convert case, Filtering – including removing stop words, Stemming)

ID	Document
Doc1	Jazz music has a swing rhythm.
Doc2	Swing is hard to explain.
Doc3	Swing rhythm is a natural rhythm.

3. Create 3 text documents following the example in the lecture notes and create a Term-Document matrix using R.

```
# set working directory to desktop
setwd("~/Desktop")
# clean up the environment before starting
rm(list = ls())
library(slam)
library(tm)
library(SnowballC)

#Get file path to folder "test" where the documents are located
cname = file.path(".", "Tute_Q3")
cname
print(dir(cname))
docs = Corpus(DirSource((cname)))
print(summary(docs))

#Tokenisation
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, content_transformer(tolower))

#Filter words
# Remove stop words and white space
docs <- tm_map(docs, removeWords, stopwords("english"))
docs <- tm_map(docs, stripWhitespace)

# Stem
docs <- tm_map(docs, stemDocument, language = "english")

#Create document term matrix
dtm <- DocumentTermMatrix(docs)
dtm = as.data.frame(as.matrix(dtm))
write.csv(dtm, "dtm.csv")
```

	explain	hard	jazz	music	natur	rhythm	swing
Doc1.txt	0	0	1	1	0	1	1
Doc2.txt	1	1	0	0	0	0	1
Doc3.txt	0	0	0	0	1	2	1

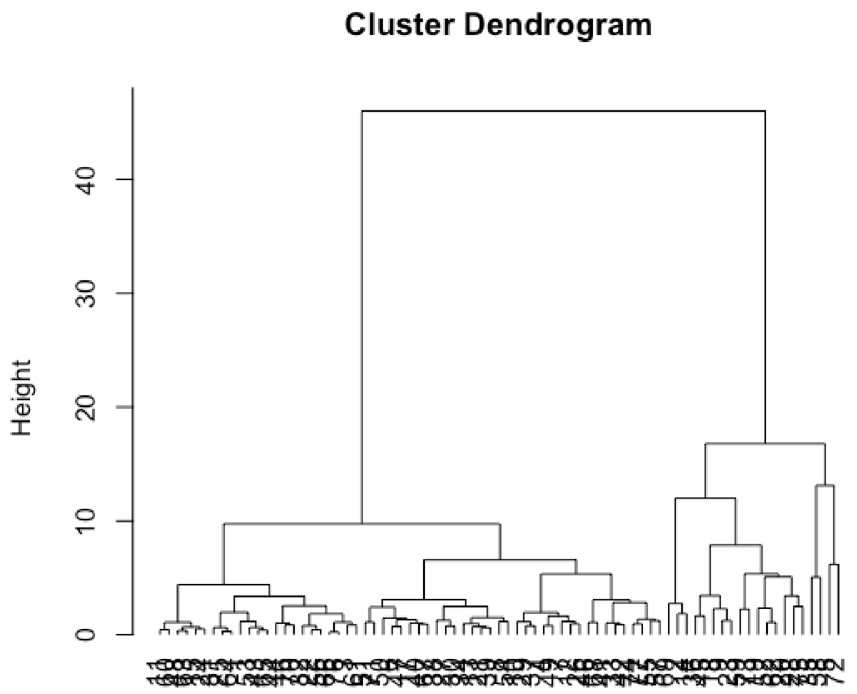
Calculate the Cosine Distance between each pair of documents using TDFM.

	explain	had	jazz	music	natur	rhythm	swing					Cosine
Doc1	0	0	1	1	0	1	1					Distance
Doc2	1	1	0	0	0	0	1					
Doc3	0	0	0	0	1	2	1					
Doc1	0	0	1	1	0	1	1		num	1.00	=	0.29
Doc2	1	1	0	0	0	0	1		den	3.46		
Doc2	1	1	0	0	0	0	1		num	1.00	=	0.24
Doc3	0	0	0	0	1	2	1		den	4.24		
Doc1	0	0	1	1	0	1	1		num	3.00	=	0.61
Doc3	0	0	0	0	1	2	1		den	4.90		

4. A number of reports relating to UFO sightings have been stored in the file (UFOsample.csv). Read the data into R, process and cluster the text. Adapt the following code fragment below to read each row of a csv file as a separate document into a corpus:

```
UFO = read.csv("UFOsample.csv", header = FALSE)
UFO = data.frame(doc_id = row.names(UFO), text =UFO[1])
colnames(UFO) = c('doc_id', 'text')
docs = Corpus(DataframeSource(UFO))
```

```
# set working directory to desktop
setwd("~/Desktop")
# clean up the environment before starting
rm(list = ls())
library(slam)
library(tm)
library(SnowballC)

#Build corpus from the text data file
UFO <- read.csv("UFOsample.csv", header = FALSE)
docs = Corpus(DataframeSource(UFO))
print(summary(docs))

#Tokenise
# Hyphen to space, ref Williams
toSpace <- content_transformer(function(x, pattern) gsub(pattern, " ", x))
docs <- tm_map(docs, toSpace, "-")
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, content_transformer(tolower))
#Filter Words
# Remove stop words and white space
docs <- tm_map(docs, removeWords, stopwords("english"))
docs <- tm_map(docs, stripWhitespace)
# Stem
docs <- tm_map(docs, stemDocument, language = "english")
#Create document term matrix
dtm <- DocumentTermMatrix(docs)
#Remove sparse terms
dim(dtm)
dtms <- removeSparseTerms(dtm, 0.2)
dtms = as.matrix(dtms)
write.csv(dtms, "dtms_UFO.csv")
#cluster
distmatrix = dist(scale(dtms))
fit = hclust(distmatrix, method = "ward.D")
plot(fit)
plot(fit, hang = -1)
```



Note very small dtm for these data – gives a better clustering

	abduct	figur	light	like	look	see	time
1	1	2	0	2	0	1	0
2	1	5	2	0	1	2	1
3	1	1	5	0	3	4	2
4	2	1	2	3	5	1	0
5	1	2	28	14	35	15	25
6	1	1	0	2	0	3	4
7	1	1	0	5	5	0	1
8	1	1	0	2	1	0	6
9	1	1	34	7	8	11	8
10	2	0	5	3	4	0	1
11	…	…	…	…	…	…	…

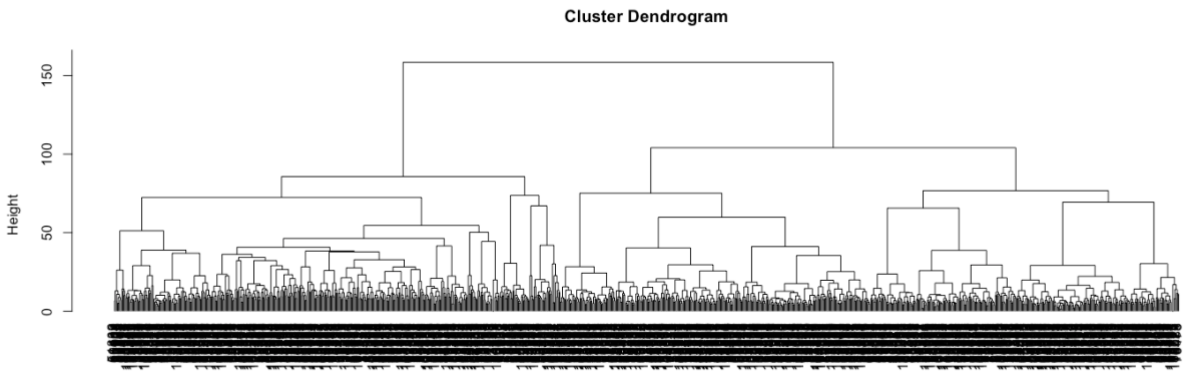
5. A selection of Usenet articles taken from 20 newsgroups are in the zipped folder (mini_newsgroups_mixedup.zip). These documents were obtained from the UCI KDD Archive of data sources for machine learning. Ref: http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.data.html. The topics are: alt.atheism; comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, comp.sys.mac.hardware, comp.windows.x, misc.forsale, rec.autos, rec.motorcycles, rec.sport.baseball, rec.sport.hockey, sci.crypt, sci.electronics, sci.med, sci.space, soc.religion.christian, talk.politics.guns, talk.politics.mideast, talk.politics.misc, talk.religion.misc.

Note: these articles have been vetted for expletives but some may contain offensive content. Therefore you are not encouraged to read these in detail. (Newsgroup topic appears in header)

(a) Process and cluster the data. Inspect the Term-Document Matrix or word frequency counts. This should identify words that appear commonly in each article, regardless of topic.

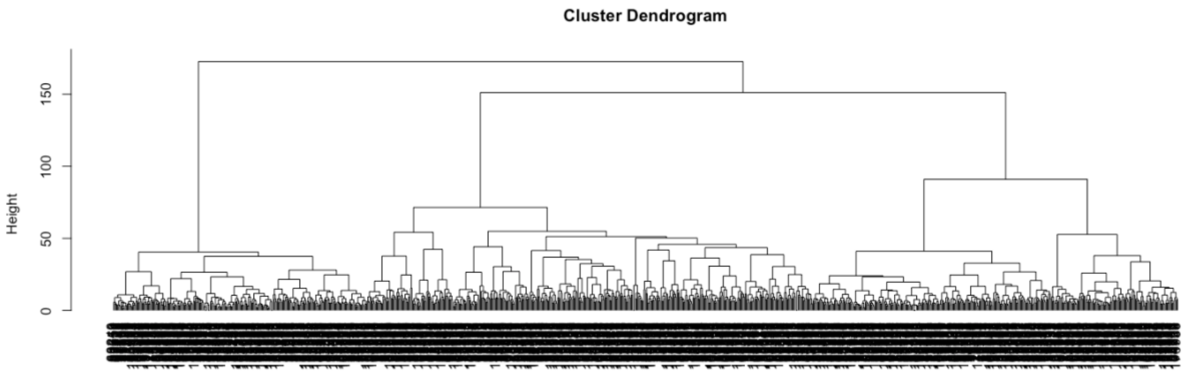
```
#Build corpus on usenet data
cname = file.path(".", "mini_newsgroups_mixedup")
docs = Corpus(DirSource((cname)))
print(summary(docs))

# Specific transformations
# Hyphen to space, ref Williams
toSpace <- content_transformer(function(x, pattern) gsub(pattern, " ", x))
docs <- tm_map(docs, toSpace, "-")
# cantaloupe to space, ref Williams
toSpace <- content_transformer(function(x, pattern) gsub(pattern, " ", x))
docs <- tm_map(docs, toSpace, "cantaloupe")
# newsgroup to space, ref Williams
toSpace <- content_transformer(function(x, pattern) gsub(pattern, " ", x))
docs <- tm_map(docs, toSpace, "newsgroup")

# Tokenise
#inspect(docs[1])
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, content_transformer(tolower))
# Remove stop words and white space
docs <- tm_map(docs, removeWords, stopwords("english"))
docs <- tm_map(docs, stripWhitespace)
# Stem
docs <- tm_map(docs, stemDocument, language = "english")

#Create document term matrix
dtm <- DocumentTermMatrix(docs)

# Check Word frequencies, ref Williams
freq <- colSums(as.matrix(dtm))
length(freq)
ord = order(freq)
freq[head(ord)]
freq[tail(ord)]

# Frequency of frequencies, ref Williams
head(table(freq), 10)
tail(table(freq), 10)
dim(dtm)
dtms <- removeSparseTerms(dtm, 0.9)
dim(dtms)

# inspect(dtms)
findFreqTerms(dtm, lowfreq = 10)
dtms = as.matrix(dtms)
write.csv(dtms, "dtms.csv")

#cluster
distmatrix = dist(scale(dtms))
fit = hclust(distmatrix, method = "ward.D")
cutfit = cutree(fit, k = 20)
plot(fit)
plot(fit, hang = -1)
```

```
cutfit
sort(cutfit)
```

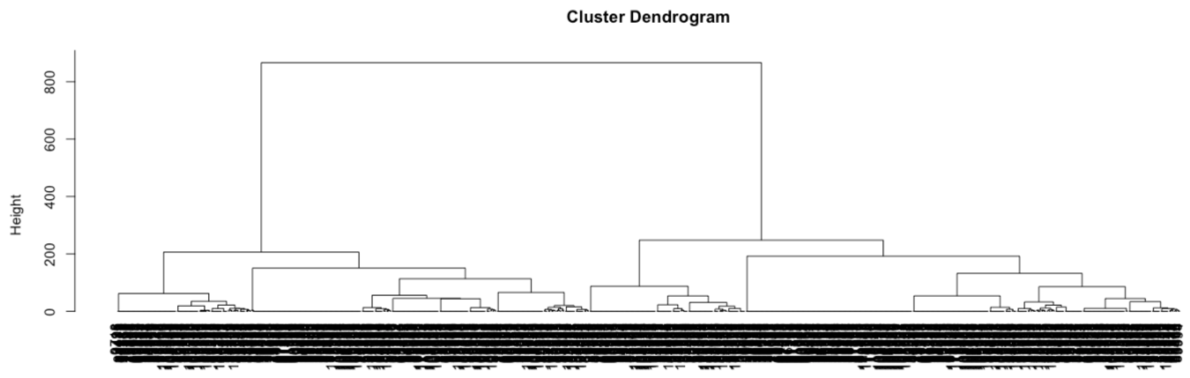


(b) Using the information gained in Part (a) re-process the articles to eliminate these specific words or phrases and then re-cluster the data. If you are using hierarchical clustering, partition the data into 20 clusters.

```
# after checking TDM a few more stems to remove
toSpace <- content_transformer(function(x, pattern) gsub(pattern, " ", x))
docs <- tm_map(docs, toSpace, "newsgroup")
docs <- tm_map(docs, toSpace, "subject")
docs <- tm_map(docs, toSpace, "messag")
docs <- tm_map(docs, toSpace, "date")
docs <- tm_map(docs, toSpace, "line")
docs <- tm_map(docs, toSpace, "organ")
docs <- tm_map(docs, toSpace, "gmt")
docs <- tm_map(docs, toSpace, "path")
docs <- tm_map(docs, toSpace, "univers")
docs <- tm_map(docs, toSpace, "refer")
docs <- tm_map(docs, toSpace, "srvcscmuedu")
docs <- tm_map(docs, toSpace,
"srvcscmuedumagnesiumclubcccmuedunewsseicmueducisohio")
docs <- tm_map(docs, toSpace, "sender")
docs <- tm_map(docs, toSpace, "usenet")
docs <- tm_map(docs, toSpace, "magnesiumclubcccmuedunewsseicmueducisohio")
docs <- tm_map(docs, toSpace, "apr")
docs <- tm_map(docs, toSpace, "write")
docs <- tm_map(docs, stripWhitespace)
```

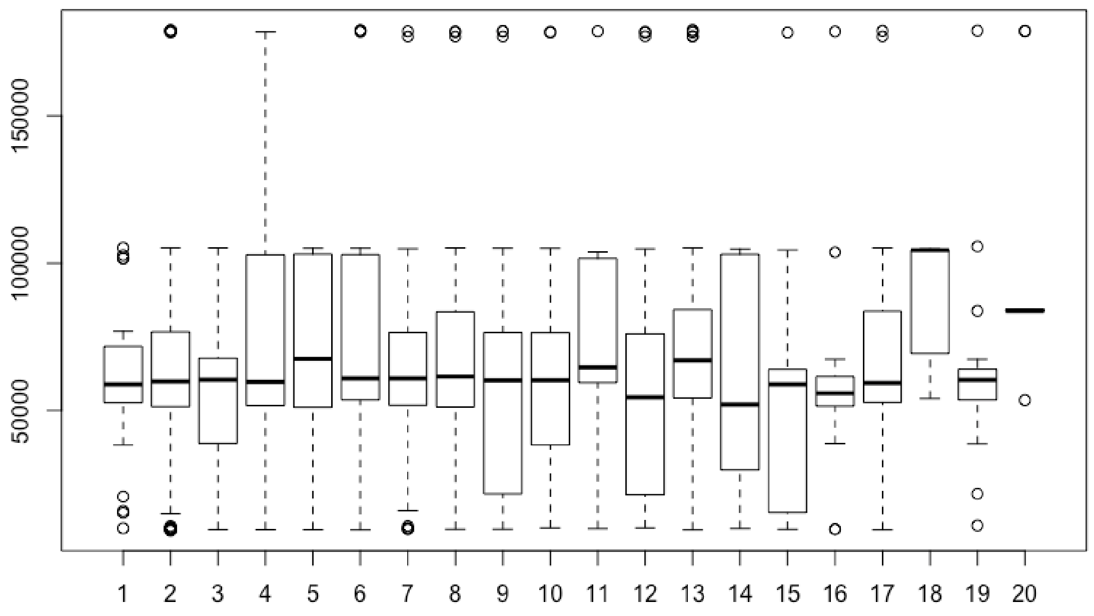


```
dtms <- removeSparseTerms(dtm, 0.9)
```



(c) Inspect the clusters obtained in Part (b), do the clusters contain articles on the same or similar topics? Hint: look at the document IDs. Newsgroup topics are coded (approximately) by topic as TTTxxx where TTT is the topic code.

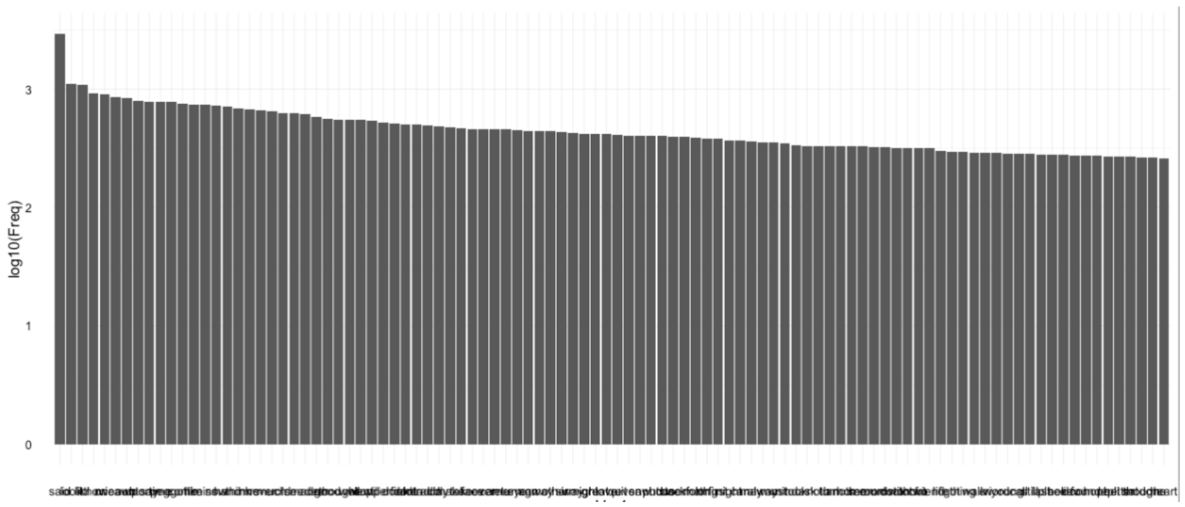
```
# I tried this, not that successful...
cutfit = as.data.frame(as.table(cutfit))
boxplot(as.numeric(as.character(Var1)) ~ Freq, data = cutfit)
```



6. The text for the novel *David Copperfield*, by Charles Dickens was obtained from Project Gutenberg, ref: https://www.gutenberg.org. An edited version of this text – with material not in the original book removed is in text file (David Copperfield PG Edit.txt).

(a) Process the data for text analysis and create a data frame of the top 100 or 200 most frequently appearing words. Plot a column graph of these word frequencies. What do you observe?

```
#Get corpus
cname = file.path(".", "David_Copperfield")
print(dir(cname))
docs = Corpus(DirSource((cname)))

# Tokenise
#writeLines(as.character(docs[[1]]))
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, content_transformer(tolower))
# Remove stop words and white space
docs <- tm_map(docs, removeWords, stopwords("english"))
docs <- tm_map(docs, stripWhitespace)
# Stem
docs <- tm_map(docs, stemDocument, language = "english")

# Create document term matrix
dtm <- DocumentTermMatrix(docs)

# Check word frequencies, ref Williams
freq <- colSums(as.matrix(dtm))
length(freq)
ord = order(freq)
freq[head(ord)]
freq[tail(ord)]

# Frequency of frequencies, ref Williams
head(table(freq), 10)
tail(table(freq), 10)
dim(dtm)

freq = as.data.frame(as.table(freq))
nfreq <- freq[order(-freq$Freq),]
nfreq = nfreq[1:100,]

#plot column graph of frequent words
nfreq$Var1 <- factor(nfreq$Var1, levels = nfreq$Var1[order(-nfreq$Freq)])
ggplot(data=nfreq, aes(x=Var1, y=log10(Freq))) + geom_bar(stat="identity")
+ theme_minimal()
```



```
print(nfreq$Var1)

  [1] said        look        littl       know        one
  [6] micawb      aunt        upon        say         time
 [11] peggotti    come        like        miss        now
 [16] hand        think       mrs         never       much
 [21] old         see         made        dear        good
 [26] thought     well        went        will        copperfield
 [31] dora        dont        head        traddl      day
 [36] make        take        face        ever        came
 [41] return      eye         agn         away        mother
 [46] sir         way         might       great       love
 [51] quit        even        saw         put         hous
 [56] back        steerforth  man         long        first
 [61] night       can         may         alway       must
```

```
[66] mind       took       ask        noth       turn
[71] anoth      home       seem       room       murdston
[76] doctor     without    two        friend     life
[81] got        thing      walk       cri        word
[86] young      call       still      last       place
[91] believ     door       found      hope       repli
[96] better     sat        though     done       heart
```

(b) Using the data in Part (a) inspect the data frame and using Wikipedia or any other source identify the main characters listed. Reprocess your data to selectively remove these characters and re-plot the word frequencies.

I'll leave this to you!