ASDM Workshop Week 5: Text Mining with R

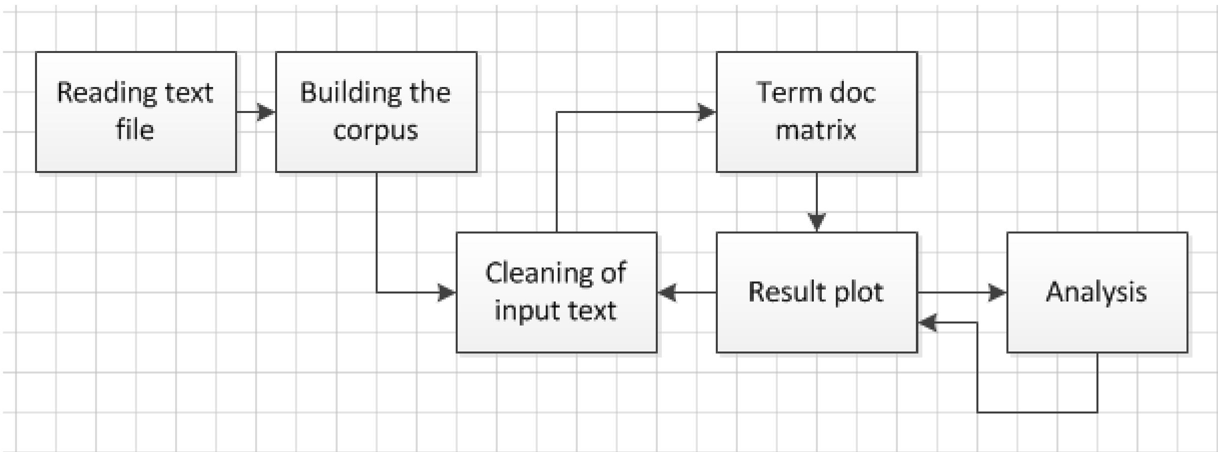
- Charith Silva, Dr. Mo Saraee

Introduction :

Text mining encompasses a vast field of theoretical approaches and methods with one thing in common: text as input information. This allows various definitions, ranging from an extension of classical data mining to texts to more sophisticated formulations like "the use of large online text collections to discover new facts and trends about the world itself" (Hearst 1999). In general, text mining is an interdisciplinary field of activity amongst data mining, linguistics, computational statistics, and computer science. Standard techniques are text classification, text clustering, ontology and taxonomy creation, document summarization and latent corpus analysis. In addition, a lot of techniques from related fields like information retrieval are commonly used (Feinerer et al., 2008).

Text mining process:

Text mining involves a series of activities to be performed in order to efficiently mine the information.



- Reading text file
- Building the corpus
- Cleaning of input text: eg : remove "commas, web links etc"
- Term doc matrix: Since there are so many words in our document how often each word appears we create a matrix for that.
- Result plot: Then we plot the results and go back and forth between analysis, plot and cleaning text.
- Analysis: Analyze the results.

Part 1: Exercise – Text mining with unstructured text data

In this workshop, we will be using “tm” package.

1. Start RStudio.

2. Change working directory.

   `File → More → Go To Working Directory…`

   In the Go To Working Directory dialogue, navigate to and select the folder where you saved your data file eg: `F:\ASDM\Week5`. Click OK.

3. Open a new R script window:
   `File → New File → R script`

4. Install “tm” package in RStudio.

```
#tm package provides a framework for text mining applications
install.packages(“tm”)
```

5. Activate tm library
```
library(tm)
```

6. Download and Read the data file

   We will be using `TMwithR.txt` data to mine text in the text file. The file can be downloaded from Blackboard.

```
dataset<- readLines("TMwithR.txt")
```

   The data is read into the data frame called “dataset”. To see what is in the document use `dataset` command.

```
dataset
```

   You can see there are 424 different items of text.

7. Inspect the dataset in R.

   Once the file has been imported to R, we often want to do few things to explore the dataset:

```
names(dataset)
head(dataset)
tail(dataset)
summary(dataset)
str(dataset)
```

8. Use the following code to create corpus.

```
#converting the text file to corpus
#Corpus is collections of documents containing (natural language) text. Corpus is the main structure that tm uses for storing and manipulating text documents.

mycorpus <- Corpus(VectorSource(dataset))
mycorpus
```

Now use the `mycorpus` command to see the documents in corpus.

Result:

```
> mycorpus
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 424
```

9. Use the following commands to inspect the items in corpus.

```
inspect(mycorpus[1])
inspect(mycorpus[2])
inspect(mycorpus[3])
```

```
> inspect(mycorpus[1])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 1

[1] IEEE TRANSACTIONS ON RELIABILITY, VOL. 58, NO. 4, DECEMBER 2009
> inspect(mycorpus[2])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 1

[1]
> inspect(mycorpus[3])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 1

[1] 649
```

```
inspect(mycorpus[8])
mycorpus[8]
```

```
> inspect(mycorpus[8])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 1

[1] Abstract--Although the need for collecting warranty data originated from financial reas
ons, it is also extensively used for modeling and analysis to support managerial decision-m
aking in industries. Strategic, tactical, and operational level decisions involving warrant
y cost very often use warranty spending forecasts that are developed using statistical meth
ods. Existing literature provides warranty forecasting approaches involving variables such
as mileage accumulation rate, failure rate, repeat repair rate, and cost per repair. Howeve
r, there are several key failure modes that are known to be influenced by seasonality. For
example, `engine slow to start' conditions drive a higher claim rate in colder months than
in warmer months. Accommodation of such failure modes influenced by seasonality has not bee
n considered in the warranty cost modeling literature. This paper presents a flexible appro
ach for developing a monthly warranty spend forecasting model that incorporates calendar mo
nth seasonality, business days per month for authorized service centers, and sales ramp-up
in addition to the earlier mentioned variables. On one hand, the model allows development o
f warranty spend forecasts for entire warranty coverage to support strategic level decision
s; on the other hand, forecasts for monthly warranty spend help support tactical and operat
ional level decisions. The workability of the proposed methodology is illustrated using an
application example.
> mycorpus[8]
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 1
```

10. Use the following command to clean up and convert the document in lower alphabets.

```
#tm_map function can be used as interface to apply
transformation functions to corpus.

mycorpus <- tm_map(mycorpus,tolower)
inspect(mycorpus[8])
```

Result:
The document has been converted to lowercase.

```
> mycorpus = tm_map(mycorpus,tolower)
> inspect(mycorpus[8])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 1

[1] abstract--although the need for collecting warranty data originated from financial reas
ons, it is also extensively used for modeling and analysis to support managerial decision-m
aking in industries. strategic, tactical, and operational level decisions involving warrant
y cost very often use warranty spending forecasts that are developed using statistical meth
ods. existing literature provides warranty forecasting approaches involving variables such
as mileage accumulation rate, failure rate, repeat repair rate, and cost per repair. howeve
r, there are several key failure modes that are known to be influenced by seasonality. for
example, `engine slow to start' conditions drive a higher claim rate in colder months than
in warmer months. accommodation of such failure modes influenced by seasonality has not bee
n considered in the warranty cost modeling literature. this paper presents a flexible appro
ach for developing a monthly warranty spend forecasting model that incorporates calendar mo
nth seasonality, business days per month for authorized service centers, and sales ramp-up
in addition to the earlier mentioned variables. on one hand, the model allows development o
f warranty spend forecasts for entire warranty coverage to support strategic level decision
s; on the other hand, forecasts for monthly warranty spend help support tactical and operat
ional level decisions. the workability of the proposed methodology is illustrated using an
application example.
```

11. Use `getTransformations()` function to retrieve the list of predefined transformations (mappings) which can be used with **tm_map** function.

```
getTransformations()
```

```
> getTransformations()
[1] "removeNumbers"     "removePunctuation" "removeWords"       "stemDocument"      "stripWhitespace"
```

12. Use the following command to remove punctuations.

```
mycorpus <-tm_map(mycorpus,removePunctuation)
inspect(mycorpus[8])
```

Result:

```
> mycorpus=tm_map(mycorpus,removePunctuation)
> inspect(mycorpus[8])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 1

[1] abstractalthough the need for collecting warranty data originated from financial reason
s it is also extensively used for modeling and analysis to support managerial decisionmakin
g in industries strategic tactical and operational level decisions involving warranty cost
very often use warranty spending forecasts that are developed using statistical methods exi
sting literature provides warranty forecasting approaches involving variables such as milea
ge accumulation rate failure rate repeat repair rate and cost per repair however there are
several key failure modes that are known to be influenced by seasonality for example engine
 slow to start conditions drive a higher claim rate in colder months than in warmer months
accommodation of such failure modes influenced by seasonality has not been considered in th
e warranty cost modeling literature this paper presents a flexible approach for developing
a monthly warranty spend forecasting model that incorporates calendar month seasonality bus
iness days per month for authorized service centers and sales rampup in addition to the ear
lier mentioned variables on one hand the model allows development of warranty spend forecas
ts for entire warranty coverage to support strategic level decisions on the other hand fore
casts for monthly warranty spend help support tactical and operational level decisions the
workability of the proposed methodology is illustrated using an application example
```

Similarly we can remove numbers using following line of code.

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

13. We can also remove stop words from the document.

Use the following line of code to see the stop words in English language.

```
stopwords("en")
```

```
> stopwords("en")
  [1] "i"          "me"         "my"         "myself"     "we"         "our"
  [7] "ours"       "ourselves"  "you"        "your"       "yours"      "yourself"
 [13] "yourselves" "he"         "him"        "his"        "himself"    "she"
 [19] "her"        "hers"       "herself"    "it"         "its"        "itself"
 [25] "they"       "them"       "their"      "theirs"     "themselves" "what"
 [31] "which"      "who"        "whom"       "this"       "that"       "these"
 [37] "those"      "am"         "is"         "are"        "was"        "were"
 [43] "be"         "been"       "being"      "have"       "has"        "had"
 [49] "having"     "do"         "does"       "did"        "doing"      "would"
 [55] "should"     "could"      "ought"      "i'm"        "you're"     "he's"
 [61] "she's"      "it's"       "we're"      "they're"    "i've"       "you've"
 [67] "we've"      "they've"    "i'd"        "you'd"      "he'd"       "she'd"
 [73] "we'd"       "they'd"     "i'll"       "you'll"     "he'll"      "she'll"
 [79] "we'll"      "they'll"    "isn't"      "aren't"     "wasn't"     "weren't"
 [85] "hasn't"     "haven't"    "hadn't"     "doesn't"    "don't"      "didn't"
 [91] "won't"      "wouldn't"   "shan't"     "shouldn't"  "can't"      "cannot"
 [97] "couldn't"   "mustn't"    "let's"      "that's"     "who's"      "what's"
[103] "here's"     "there's"    "when's"     "where's"    "why's"      "how's"
[109] "a"          "an"         "the"        "and"        "but"        "if"
[115] "or"         "because"    "as"         "until"      "while"      "of"
[121] "at"         "by"         "for"        "with"       "about"      "against"
[127] "between"    "into"       "through"    "during"     "before"     "after"
[133] "above"      "below"      "to"         "from"       "up"         "down"
[139] "in"         "out"        "on"         "off"        "over"       "under"
[145] "again"      "further"    "then"       "once"       "here"       "there"
[151] "when"       "where"      "why"        "how"        "all"        "any"
[157] "both"       "each"       "few"        "more"       "most"       "other"
[163] "some"       "such"       "no"         "nor"        "not"        "only"
```

Use the following line of code to remove stop words:

```
dataclean <-tm_map(mycorpus,removeWords,stopwords("english"))
inspect(dataclean[8])
```

Result:

```
> dataclean <-tm_map(mycorpus,removeWords,stopwords("english"))
> inspect(dataclean[8])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 1

[1] abstractalthough  need  collecting warranty data originated  financial reasons   also e
xtensively used  modeling  analysis  support managerial decisionmaking  industries strategi
c tactical  operational level decisions involving warranty cost  often use warranty spendin
g forecasts   developed using statistical methods existing literature provides warranty for
ecasting approaches involving variables   mileage accumulation rate failure rate repeat rep
air rate  cost per repair however   several key failure modes  known   influenced  seasona
lity  example engine slow  start conditions drive  higher claim rate  colder months   warme
r months accommodation   failure modes influenced  seasonality   considered   warranty cos
t modeling literature  paper presents  flexible approach  developing  monthly warranty spen
d forecasting model  incorporates calendar month seasonality business days per month  autho
rized service centers  sales rampup  addition   earlier mentioned variables  one hand  mode
l allows development  warranty spend forecasts  entire warranty coverage  support strategic
 level decisions    hand forecasts  monthly warranty spend help support tactical  operation
al level decisions  workability   proposed methodology  illustrated using  application exam
ple
```

14. As you can see there are lots of white spaces in the document. Use the following line of code to remove those white spaces.

```
dataclean <- tm_map(dataclean,stripWhitespace)
inspect(dataclean[8])
```

Result:

```
> dataclean <- tm_map(dataclean,stripWhitespace)
> inspect(dataclean[8])
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 1

[1] abstractalthough need collecting warranty data originated financial reasons also extens
ively used modeling analysis support managerial decisionmaking industries strategic tactica
l operational level decisions involving warranty cost often use warranty spending forecasts
 developed using statistical methods existing literature provides warranty forecasting appr
oaches involving variables mileage accumulation rate failure rate repeat repair rate cost p
er repair however several key failure modes known influenced seasonality example engine slo
w start conditions drive higher claim rate colder months warmer months accommodation failur
e modes influenced seasonality considered warranty cost modeling literature paper presents
flexible approach developing monthly warranty spend forecasting model incorporates calendar
 month seasonality business days per month authorized service centers sales rampup addition
 earlier mentioned variables one hand model allows development warranty spend forecasts ent
ire warranty coverage support strategic level decisions hand forecasts monthly warranty spe
nd help support tactical operational level decisions workability proposed methodology illus
trated using application example
```

15. The next step is to create a Document-Term Matrix (DTM). DTM is a matrix that lists all occurrences of words in the corpus. In DTM, documents are represented by rows and the terms (or words) by columns. If a word occurs in a particular document n times, then the matrix entry for corresponding to that row and column is n, if it doesn't occur at all, the entry is 0.

    Use the following line of code to create the term document matri

    ```
    dtm <- TermDocumentMatrix(dataclean,
                              control = list(minWordLength=c(1,Inf))
                              )
    dtm
    ```

    ```
    > dtm
    <<TermDocumentMatrix (terms: 718, documents: 424)>>
    Non-/sparse entries: 2406/302026
    Sparsity           : 99%
    Maximal term length: 22
    Weighting          : term frequency (tf)
    ```

    ```
    #findFreqTerms function can be used to find frequent terms in a
    document-term or term-document matrix.

    findFreqTerms(dtm,lowfreq = 2)
    ```

    Result:

    ```
    > dtm=TermDocumentMatrix(dataclean,control = list(minWordLength=c(1,Inf)))
    > findFreqTerms(dtm,lowfreq = 2)
     [1] "2009"          "december"      "ieee"          "reliability"
     [5] "transactions"  "vol"           "calendar"      "failures"
     [9] "forecasting"   "influenced"    "month"         "seasonality"
    [13] "spend"         "subsystem"     "warranty"      "rai"
    [17] "accumulation"  "addition"      "also"          "analysis"
    [21] "application"   "approach"      "approaches"    "authorized"
    [25] "business"      "centers"       "claim"         "conditions"
    [29] "cost"          "coverage"      "data"          "days"
    [33] "decisions"     "developed"     "developing"    "engine"
    [37] "example"       "extensively"   "failure"       "flexible"
    [41] "forecasts"     "hand"          "help"          "higher"
    [45] "however"       "illustrated"   "incorporates"  "involving"
    [49] "key"           "known"         "level"         "literature"
    [53] "methodology"   "methods"       "mileage"       "model"
    [57] "modeling"      "modes"         "monthly"       "months"
    [61] "need"          "often"         "one"           "operational"
    [65] "paper"         "per"           "presents"      "proposed"
    [69] "provides"      "rampup"        "rate"          "repair"
    ```

16. Use the following line of code to see the words with frequency in the document matrix.

    ```
    termFrequency <-  rowSums(as.matrix(dtm))
    termFrequency
    ```

Result:

```
> termFrequency = rowSums(as.matrix(dtm))
> termFrequency
           2009        december            ieee      reliability     transactions
             16              15              14                8                5
            vol             649        calendar         failures      forecasting
              5               1               7                5               21
     influenced           month     seasonality            spend        subsystem
             15              75              32                9               15
       warranty     bharatendra             rai abstractalthough    accommodation
            120               1               8                1                1
   accumulation        addition          allows             also         analysis
             15               2               1                9                5
    application        approach      approaches       authorized         business
              4               4               3               12               35
        centers           claim          colder       collecting       conditions
              2              33               1                1                3
     considered            cost        coverage             data             days
              1              46               8               18               34
 decisionmaking       decisions       developed       developing      development
              1              18               8                7                1
```

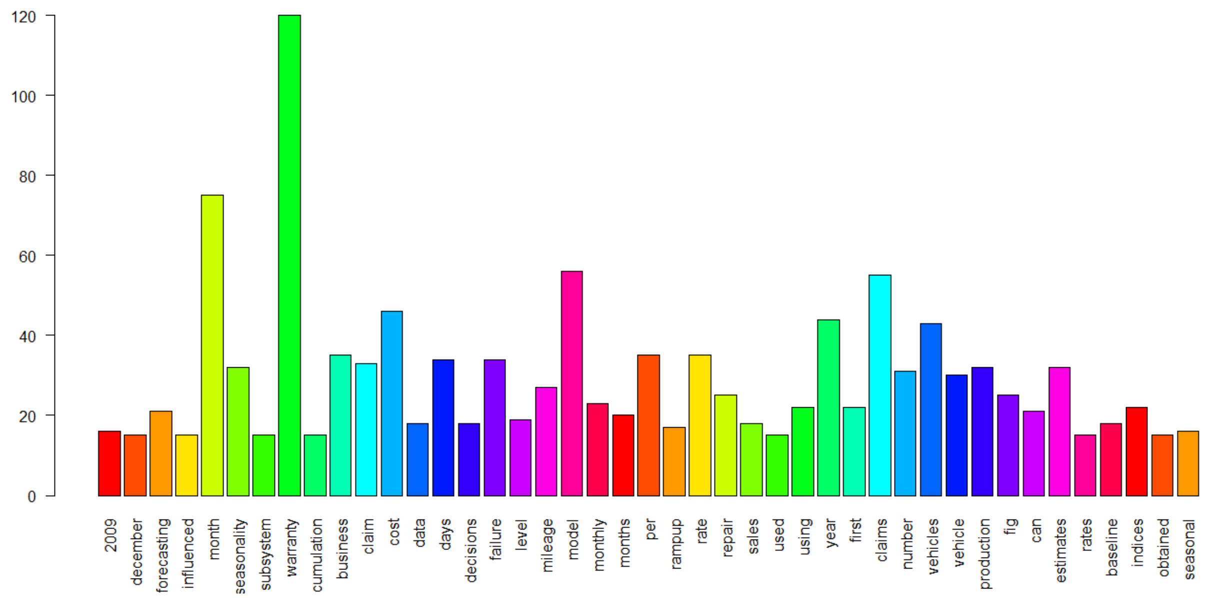
Use the following line of code to see the words with frequency greater than 15.

```
termFrequency <- subset(termFrequency,termFrequency>=15)
termFrequency
```

```
> termFrequency=subset(termFrequency,termFrequency>=15)
> termFrequency
        2009     december  forecasting   influenced        month  seasonality    subsystem     warranty
          16           15           21           15           75           32           15          120
accumulation     business        claim         cost         data         days    decisions      failure
          15           35           33           46           18           34           18           34
       level      mileage        model      monthly       months          per       rampup         rate
          19           27           56           23           20           35           17           35
      repair        sales         used        using         year        first       claims       number
          25           18           15           22           44           22           55           31
    vehicles      vehicle   production          fig          can    estimates        rates     baseline
          43           30           32           25           21           32           15           18
     indices     obtained     seasonal
          22           15           16
```

17. Plot results with barplot() function

```
barplot(termFrequency,las=2,col=rainbow(20))
```



18. Finally to see the world cloud of the words, use the “wordcloud” package.
URL: https://cran.r-project.org/web/packages/wordcloud/wordcloud.pdf

```
install.packages("wordcloud")    #install "wordcloud"
library(wordcloud)              #load "wordcloud"
```

Use following lines of code to see the word cloud.

```
wordfreq <-sort(termFrequency,decreasing = TRUE)
wordfreq
```

```
> wordfreq<-sort(termFrequency,decreasing = TRUE)
> wordfreq
       warranty          month          model         claims
            120             89             56             55
           cost           year       vehicles       business
             46             44             43             35
            per           rate           days        failure
             35             35             34             34
          claim    seasonality     production      estimates
             33             32             32             32
         number        vehicle        mileage         repair
             31             30             27             25
            fig        monthly          using          first
             25             23             22             22
        indices    forecasting            can         months
             22             21             21             20
```

Descripton of the **`wordcloud()`** function

Description

Plot a word cloud

Usage

```
wordcloud(words,freq,scale=c(4,.5),min.freq=3,max.words=Inf,
random.order=TRUE, random.color=FALSE, rot.per=.1,
colors="black",ordered.colors=FALSE,use.r.layout=FALSE,
fixed.asp=TRUE, ...)
```

Arguments

`words`	the words
`freq`	their frequencies
`scale`	A vector of length 2 indicating the range of the size of the words.
`min.freq`	words with frequency below min.freq will not be plotted
`max.words`	Maximum number of words to be plotted. least frequent terms dropped
`random.order`	plot words in random order. If false, they will be plotted in decreasing frequency
`random.color`	choose colors randomly from the colors. If false, the color is chosen based on the frequency
`rot.per`	proportion words with 90 degree rotation
`colors`	color words from least to most frequent
`ordered.colors`	if true, then colors are assigned to words in order
`use.r.layout`	if false, then c++ code is used for collision detection, otherwise R is used
`fixed.asp`	if TRUE, the aspect ratio is fixed. Variable aspect ratio only supported if rot.per==0
`...`	Additional parameters to be passed to text (and strheight,strwidth).

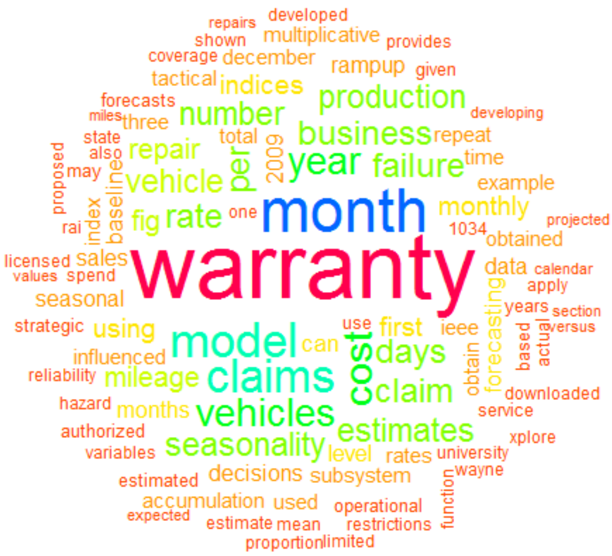
Details

If freq is missing, then words can either be a character vector, or Corpus. If it is a vector and freq is missing, standard stop words will be removed prior to plotting.

```
wordcloud(words = names(wordfreq),
          freq=wordfreq,max.words=100,
          min.freq = 5,
          random.order = F)
```

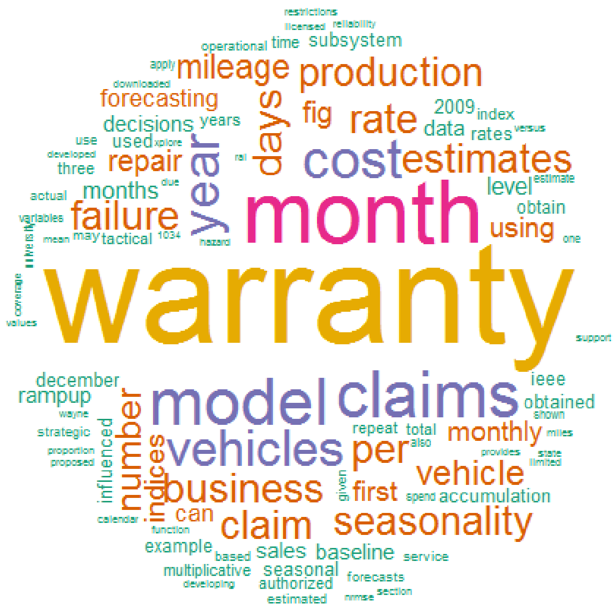
There are many different variations of word cloud.

```
wordcloud(words = names(wordfreq),
          freq=wordfreq,max.words=100,
          min.freq = 5,
          random.order = F,
          colors = rainbow(20))
```



```
wordcloud(words = names(wordfreq),
          freq=wordfreq,max.words=100,
          min.freq = 5,
          random.order = F,
          colors = brewer.pal(6,"Dark2"))
```

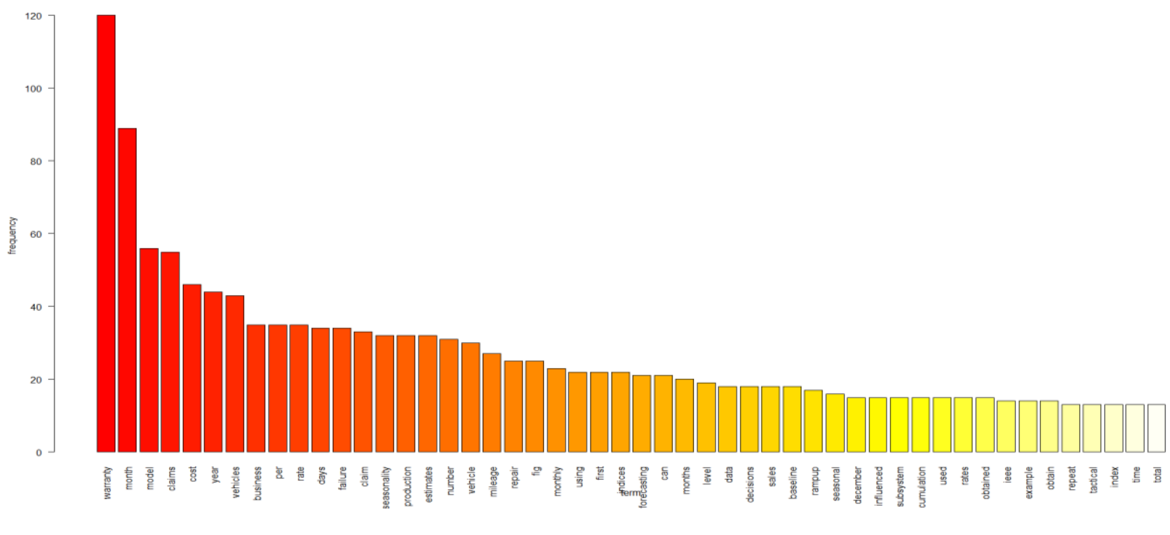

```
wordcloud(words = names(wordfreq),
          scale = c(6,.05),
          freq=wordfreq,max.words=100,
          min.freq = 5,
          random.order = F,
          colors = brewer.pal(6,"Dark2"))
```



```
wordcloud(words = names(wordfreq),
          rot.per=0.50,
          scale = c(6,.05),
          freq=wordfreq,
          max.words=100,
          min.freq = 5,
          random.order = F,
          colors = brewer.pal(6,"Dark2"))
```

19. Word distribution

```
#We now see the distribution of the 50 most frequent words in a
barplot.

#We now see the distribution of the 50 most frequent words in a
barplot.
barplot(wordfreq[1:50],
        xlab = "term",
        ylab = "frequency",
        las=2,
        col=heat.colors(50))
```



Part 2: Exercise - Text mining and clustering using Twitter data

In this tutorial, we will use Twitter data to compare the trend of popular health related news from 3 news agencies.

Twitter is a popular micro blogging service where users create status messages (called "tweets"). Tweets are short messages with a maximum length of 140 characters. The distinguishing characteristics of tweets are hashtags. Hashtags are used for logically grouping tweets and searching them. Authors of those messages write about their life, share opinions on variety of topics and discuss current issues. As more and more users post about products and services they use, or express their political, religious views, etc… Micro blogging websites become valuable sources of people's opinions and sentiments. Such data can be efficiently used for marketing or social studies. As a result, there has been a tremendous need to design methods and algorithms which can effectively process a wide variety of text applications (source : Suprajha S,Yogitha C :A Study on Sentiment Analysis using Tweeter Data).

Dataset Information:

The data was collected in 2015 using Twitter API. Each file is related to one Twitter account of a news agency. For example, bbchealth.csv is related to BBC health news. Each line of the dataset contains tweet id, date and time and tweet. This text data has been used to evaluate the performance of topic models on short text data. However, it can be used for other tasks such as clustering.

```
#Read the data file
bbchealth <- read.csv("bbchealth.csv", header= TRUE)
cnnhealth <- read.csv("cnnhealth.csv", header= TRUE)
foxhealth <- read.csv("foxnewshealth.csv", header= TRUE)
```

```
#Inspect the dataset
head(bbchealth)
head(cnnhealth)
head(foxhealth)
```

```
> head(bbchealth)
   twee_id                  date.and.time                                               tweet
1 5.86e+17 Thu Apr 09 01:31:50 +0000 2015      Breast cancer risk test devised http://bbc.in/1CimpJF
2 5.86e+17 Wed Apr 08 23:30:18 +0000 2015  GP workload harming care - BMA poll http://bbc.in/1ChTBRv
3 5.86e+17 Wed Apr 08 23:30:18 +0000 2015  Short people's 'heart risk greater' http://bbc.in/1ChTANp
4 5.86e+17 Wed Apr 08 18:05:28 +0000 2015 New approach against HIV 'promising' http://bbc.in/1E6jAjt
5 5.86e+17 Wed Apr 08 13:19:33 +0000 2015 Coalition 'undermined NHS' - doctors http://bbc.in/1CnLwK7
6 5.86e+17 Wed Apr 08 09:18:39 +0000 2015   Review of case against NHS manager http://bbc.in/1Ffj6ci
```

```
#Inspect the tweet column in the datasets
head(bbchealth$tweet)
head(cnnhealth$tweet)
```

```
head(foxhealth$tweet)


#create text vectors
bbchealth_tweet<- bbchealth$tweet
cnnhealth_tweet<- cnnhealth$tweet
foxhealth_tweet<- foxhealth$tweet


#convert all text to lower case
bbchealth_tweet<- tolower(bbchealth_tweet)
cnnhealth_tweet<- tolower(cnnhealth_tweet)
foxhealth_tweet<- tolower(foxhealth_tweet)


#gsub() function replaces all matches of a string, if the
parameter is a string vector, returns a string vector of the same
length and with the same attributes (after possible coercion to
character). Elements of string vectors which are not substituted
will be returned unchanged (including any declared encoding).
gsub() function can use regular expressions as search string.
```

What is a regular expression?

Regular expression is a pattern that describes a set of strings. Simply speaking, regular expression is an" instruction" given to a function on what and how to match or replace strings.

Additional reading materials about regular expression

- https://rstudio-pubs-static.s3.amazonaws.com/74603_76cd14d5983f47408fdf0b323550b846.html
- http://biostat.mc.vanderbilt.edu/wiki/pub/Main/SvetlanaEdenRFiles/regExprTalk.pdf

Some popular regular expression syntaxs:

Syntax	Description
\\d	Digit, 0,1,2 ... 9
\\D	Not Digit
\\s	Space
\\S	Not Space
\\w	Word
\\W	Not Word
\\t	Tab
\\n	New line
^	Beginning of the string
$	End of the string
****	Escape special characters, e.g. \\ is "\", \+ is "+"
|	Alternation match. e.g. /(e|d)n/ matches "en" and "dn"
.	Any character, except \n or line terminator

[ab]	a or b
[^ab]	Any character except a and b
[0-9]	All Digit
[A-Z]	All uppercase A to Z letters
[a-z]	All lowercase a to z letters
[A-z]	All Uppercase and lowercase a to z letters
i+	i at least one time
i*	i zero or more times
i?	i zero or 1 time
i{n}	i occurs n times in sequence
i{n1,n2}	i occurs n1 - n2 times in sequence
i{n1,n2}?	non greedy match, see above example
i{n,}	i occures >= n times
[:alnum:]	Alphanumeric characters: [:alpha:] and [:digit:]
[:alpha:]	Alphabetic characters: [:lower:] and [:upper:]
[:blank:]	Blank characters: e.g. space, tab
[:cntrl:]	Control characters
[:digit:]	Digits: 0 1 2 3 4 5 6 7 8 9
[:graph:]	Graphical characters: [:alnum:] and [:punct:]
[:lower:]	Lower-case letters in the current locale
[:print:]	Printable characters: [:alnum:], [:punct:] and space
[:punct:]	Punctuation character: ! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [\ ] ^ _ ` { | } ~
[:space:]	Space characters: tab, newline, vertical tab, form feed, carriage return, space
[:upper:]	Upper-case letters in the current locale
[:xdigit:]	Hexadecimal digits: 0 1 2 3 4 5 6 7 8 9 A B C D E F a b c d e f

```
#Replace blank space (“rt”)
bbchealth_tweet <- gsub("rt", "", bbchealth_tweet)
cnnhealth_tweet <- gsub("rt", "", cnnhealth_tweet)
foxhealth_tweet <- gsub("rt", "", foxhealth_tweet)


#Replace tweeter @UserName
bbchealth_tweet <- gsub("@\\w+", "", bbchealth_tweet)
cnnhealth_tweet <- gsub("@\\w+", "", cnnhealth_tweet)
foxhealth_tweet <- gsub("@\\w+", "", foxhealth_tweet)


#Replace links in the tweets
bbchealth_tweet <- gsub("http\\S+\\s*", "", bbchealth_tweet)
cnnhealth_tweet <- gsub("http\\S+\\s*", "", cnnhealth_tweet)
foxhealth_tweet <- gsub("http\\S+\\s*", "", foxhealth_tweet)


#Remove punctuation
bbchealth_tweet <- gsub("[[:punct:]]", "", bbchealth_tweet)
cnnhealth_tweet <- gsub("[[:punct:]]", "", cnnhealth_tweet)
foxhealth_tweet <- gsub("[[:punct:]]", "", foxhealth_tweet)
```

```
#Remove tabs
bbchealth_tweet <- gsub("[ |\t]{2,}", "", bbchealth_tweet)
cnnhealth_tweet <- gsub("[ |\t]{2,}", "", cnnhealth_tweet)
foxhealth_tweet <- gsub("[ |\t]{2,}", "", foxhealth_tweet)



#Remove "video" word in the tweets
bbchealth_tweet <- gsub("video", "", bbchealth_tweet)
cnnhealth_tweet <- gsub("video", "", cnnhealth_tweet)
foxhealth_tweet <- gsub("video", "", foxhealth_tweet)



#Remove blank spaces at the beginning
bbchealth_tweet <- gsub("^ ", "", bbchealth_tweet)
cnnhealth_tweet <- gsub("^ ", "", cnnhealth_tweet)
foxhealth_tweet <- gsub("^ ", "", foxhealth_tweet)



#Remove blank spaces at the end
bbchealth_tweet <- gsub(" $", "", bbchealth_tweet)
cnnhealth_tweet <- gsub(" $", "", cnnhealth_tweet)
foxhealth_tweet <- gsub(" $", "", foxhealth_tweet)



#Inspect the vectors after cleaning
head(bbchealth_tweet)
head(cnnhealth_tweet)
head(foxhealth_tweet)
```

```
> head(bbchealth_tweet)
[1] "breast cancer risk test devised"    "gp workload harming carebma poll"   "sho peoples hea risk greater"
[4] "new approach against hiv promising" "coalition undermined nhsdoctors"    "review of case against nhs manager"
> head(cnnhealth_tweet)
[1] "an abundance of online info can turn us into ehypochondriacs or worse lead us to neglect getting the care we need"
[2] "a plantbased diet that incorporates fish may be the key to preventing colorectal cancers"
[3] "it doesnt take much to damage your hearing at a spos bar or nightclub thats why a billion people are at risk"
[4] "forever young discover this islandâ€™s secrets to longevity on thewonderlist w"
[5] "is posttraumatic stress disorder in your genes a simple blood test may one day help tell you"
[6] "maysoon zayid a touring standup comic with cerebral palsy has a message to share"
> head(foxhealth_tweet)
[1] "injury prevention programs unpopular with high school coaches" "6 dietary changes to make midlife"
[3] "massachusetts governor gets head shaved to suppo charity"      "dad wins 3 marathons in 8 days winnings to help ailing son"
[5] "possible cure for melanoma"                                    "wear orange glasses to get better sleep study says"
```

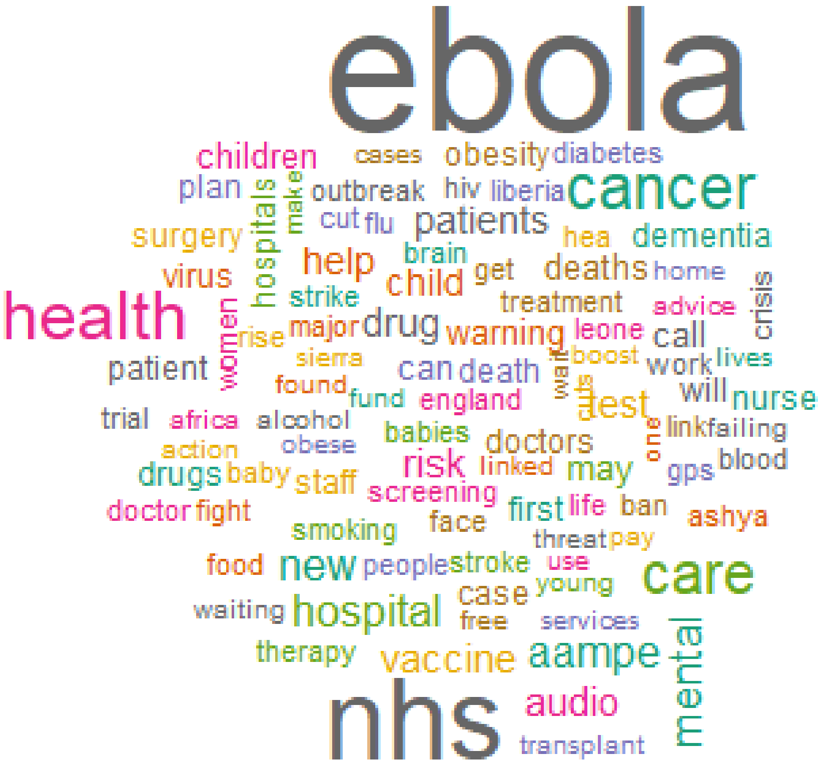
```
library(tm) #load tm package

#converting the text vectors to corpus
bbchealth_corpus <- Corpus(VectorSource(bbchealth_tweet))
bbchealth_corpus
```

```
cnnhealth_corpus <- Corpus(VectorSource(cnnhealth_tweet))
cnnhealth_corpus

foxhealth_corpus <- Corpus(VectorSource(foxhealth_tweet))
foxhealth_corpus
```

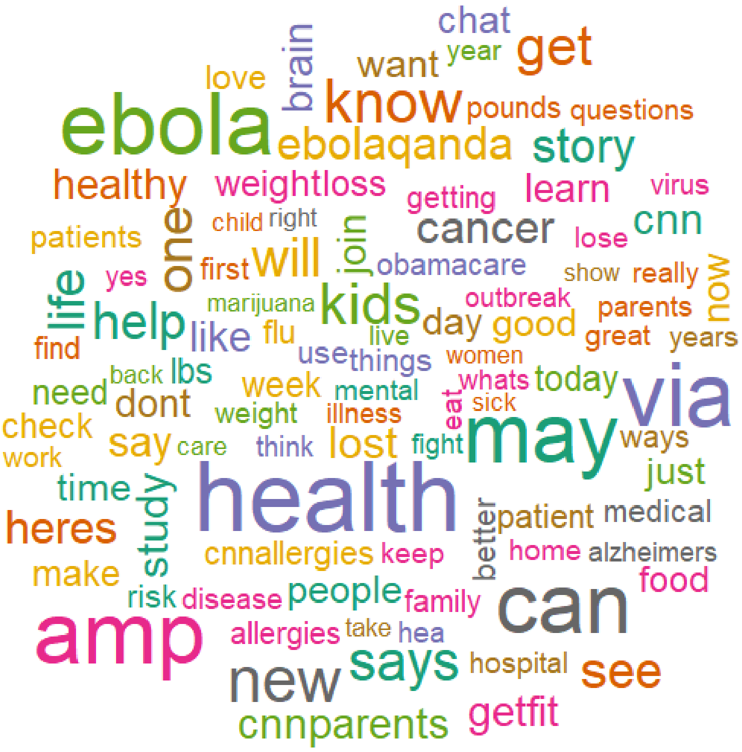
```
> bbchealth_corpus
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 2000
> cnnhealth_corpus <- Corpus(VectorSource(cnnhealth_tweet))
> cnnhealth_corpus
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 2000
> foxhealth_corpus <- Corpus(VectorSource(foxhealth_tweet))
> foxhealth_corpus
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 2000
```

```
#clean up corpus by removing stop words, number and Whitespace

bbchealth_corpus  <- tm_map(bbchealth_corpus,
                            removeWords,stopwords("english"))
bbchealth_corpus  <- tm_map(bbchealth_corpus,removeNumbers)
bbchealth_corpus  <- tm_map(bbchealth_corpus,stripWhitespace)
inspect(bbchealth_corpus )



cnnhealth_corpus <- tm_map(cnnhealth_corpus,
                           removeWords,stopwords("english"))
cnnhealth_corpus <- tm_map(cnnhealth_corpus, removeNumbers)
cnnhealth_corpus <- tm_map(cnnhealth_corpus, stripWhitespace)
inspect(cnnhealth_corpus )



foxhealth_corpus  <- tm_map(foxhealth_corpus ,
                            removeWords,stopwords("english"))
foxhealth_corpus  <- tm_map(foxhealth_corpus , removeNumbers)
foxhealth_corpus  <- tm_map(foxhealth_corpus , stripWhitespace)
inspect(foxhealth_corpus )
```

```
library("wordcloud") #load wordcloud package


#generate wordclouds
wordcloud(bbchealth_corpus,
          min.freq = 10,
          colors=brewer.pal(8, "Dark2"),
          random.color = TRUE,
          max.words = 100)
```



```
wordcloud(cnnhealth_corpus,
          min.freq = 10,
          colors=brewer.pal(8, "Dark2"),
          random.color = TRUE,
          max.words = 100)
```




```
wordcloud(foxhealth_corpus,
          min.freq = 10,
          colors=brewer.pal(8, "Dark2"),
          random.color = TRUE,
          max.words = 100)
```

```
#Create document-term matrix

bbchealth_dtm <- DocumentTermMatrix(
                      bbchealth_corpus,
                      control = list(minWordLength=c(3,Inf),
                                     bounds = list(global = c(40, Inf)))
                                     )


cnnhealth_dtm <- DocumentTermMatrix(
                      cnnhealth_corpus,
                      control = list(minWordLength=c(3,Inf),
                                     bounds = list(global = c(40, Inf)))
                                     )


foxhealth_dtm <- DocumentTermMatrix(
                      foxhealth_corpus,
                      control = list(minWordLength=c(3,Inf),
                                     bounds = list(global = c(40, Inf)))
                                     )

bbchealth_dtm
cnnhealth_dtm
foxhealth_dtm
```

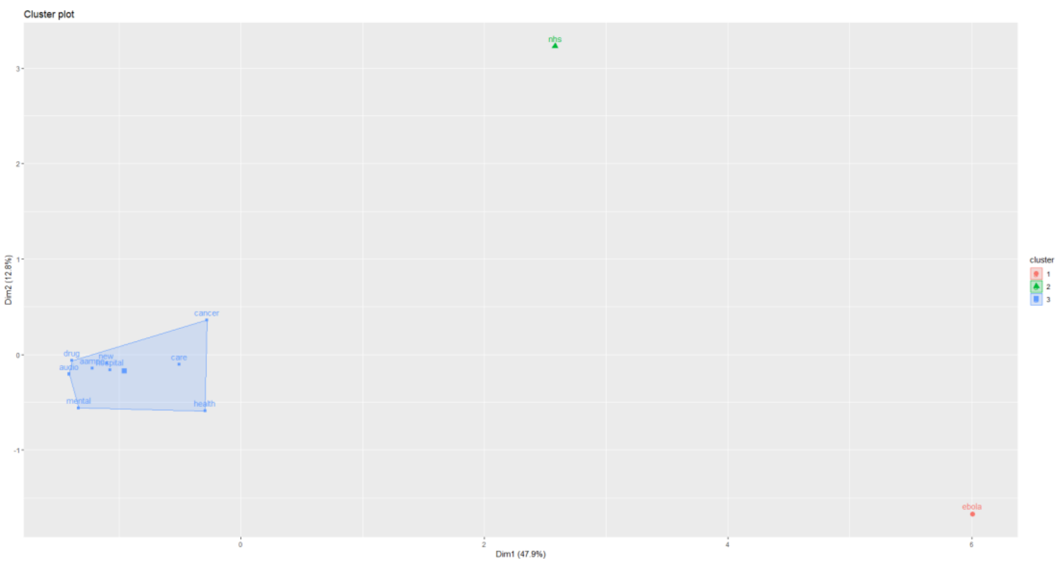
```
> bbchealth_dtm
<<DocumentTermMatrix (documents: 2000, terms: 11)>>
Non-/sparse entries: 1109/20891
Sparsity           : 95%
Maximal term length: 8
Weighting          : term frequency (tf)
> cnnhealth_dtm
<<DocumentTermMatrix (documents: 2000, terms: 33)>>
Non-/sparse entries: 2095/63905
Sparsity           : 97%
Maximal term length: 10
Weighting          : term frequency (tf)
> foxhealth_dtm
<<DocumentTermMatrix (documents: 2000, terms: 29)>>
Non-/sparse entries: 2200/55800
Sparsity           : 96%
Maximal term length: 8
Weighting          : term frequency (tf)
```

```
bbchealth_dtm2  <- as.matrix(bbchealth_dtm)
cnnhealth_dtm2  <- as.matrix(cnnhealth_dtm)
foxhealth_dtm2  <- as.matrix(foxhealth_dtm)


#K-means clustering
library(cluster)     #load cluster package
library(factoextra) #load factoextra package
```

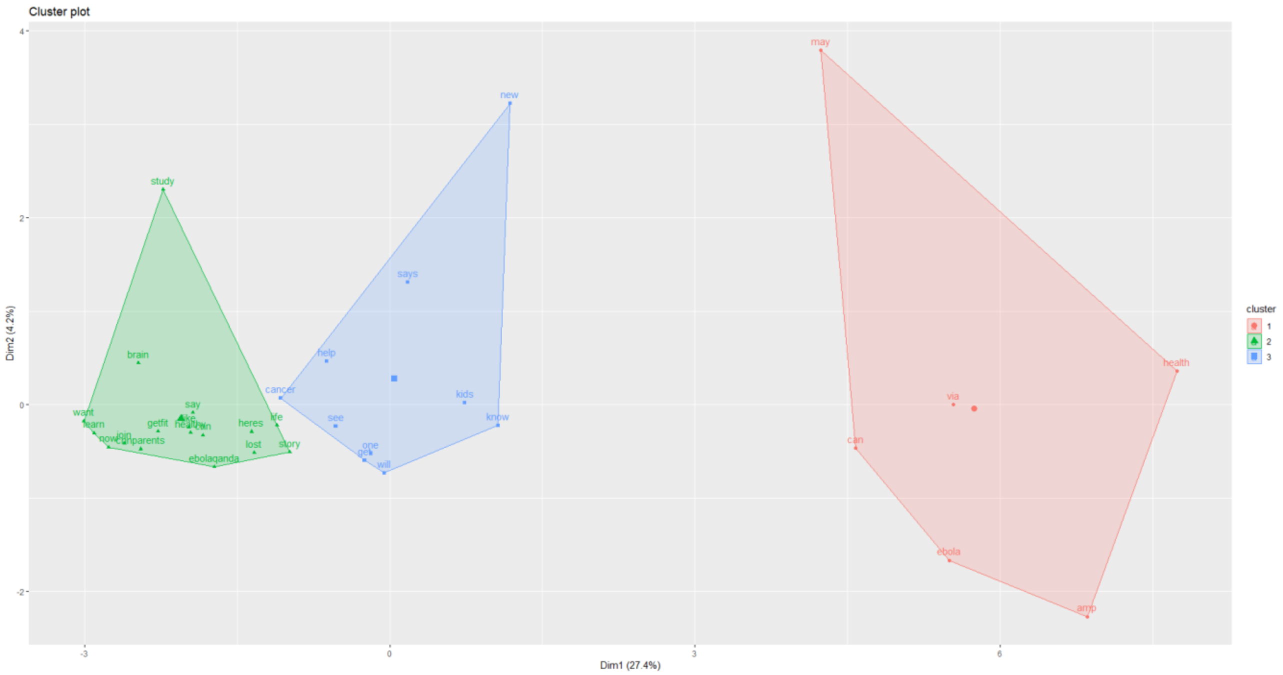
```
head(bbchealth_dtm2)
bbc_dist <- dist(t(bbchealth_dtm2), method="euclidian")
kfit <- kmeans(bbc_dist, 3)
bbc_dist
kfit
fviz_cluster(kfit,bbc_dist)
```

```
> head(bbchealth_dtm2)
    Terms
Docs cancer new nhs care aampe health hospital drug mental ebola audio
   1      1   0   0    0     0      0        0    0      0     0     0
   2      0   0   0    0     0      0        0    0      0     0     0
   3      0   0   0    0     0      0        0    0      0     0     0
   4      0   1   0    0     0      0        0    0      0     0     0
   5      0   0   0    0     0      0        0    0      0     0     0
   6      0   0   1    0     0      0        0    0      0     0     0
> bbc_dist <- dist(t(bbchealth_dtm2), method="euclidian")
> kfit <- kmeans(bbc_dist, 3)
> bbc_dist
            cancer       new       nhs      care     aampe    health  hospital      drug    mental     ebola
new      11.874342
nhs      16.401219 15.620499
care     12.961481 11.704700 16.643317
aampe    11.789826 10.000000 15.748016 11.269428
health   13.928388 12.369317 17.349352 13.190906 12.288206
hospital 11.958261 10.392305 16.000000 11.180340 10.099505 12.529964
drug     10.488088  9.539392 15.198684 11.135529  9.539392 12.000000  9.949874
mental   12.124356 10.295630 16.062378 11.180340 10.198039  7.280110 10.488088  9.949874
ebola    20.591260 19.364917 22.825424 20.248457 19.519221 20.639767 19.621417 18.973666 19.723083
audio    11.401754  9.643651 15.588457 10.954451  9.219544 12.000000  9.643651  9.055385  9.746794 19.235384
```

```
> kfit
K-means clustering with 3 clusters of sizes 1, 1, 9

Cluster means:
    cancer       new      nhs     care     aampe   health  hospital      drug    mental    ebola     audio
1 20.59126 19.364917 22.82542 20.24846 19.519221 20.63977 19.621417 18.973666 19.723083  0.00000 19.235384
2 16.40122 15.620499  0.00000 16.64332 15.748016 17.34935 16.000000 15.198684 16.062378 22.82542 15.588457
3 10.72517  9.535482 16.06799 10.39746  9.378216 10.62077  9.582443  9.073059  9.029248 19.76857  9.073915

Clustering vector:
  cancer      new      nhs     care    aampe   health hospital     drug   mental    ebola    audio
       3        3        2        3        3        3        3        3        3        1        3

Within cluster sum of squares by cluster:
[1]    0.000    0.000 1058.825
 (between_SS / total_SS =  62.0 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"    "size"
[8] "iter"         "ifault"
```



```
head(cnnhealth_dtm2)
cnn_dist <- dist(t(cnnhealth_dtm2), method="euclidian")
kfit <- kmeans(cnn_dist, 3)
cnn_dist
kfit
fviz_cluster(kfit,cnn_dist)
```




```
head(foxhealth_dtm2)
fox_dist <- dist(t(foxhealth_dtm2), method="euclidian")
kfit <- kmeans(fox_dist, 3)
fox_dist
kfit
fviz_cluster(kfit,fox_dist)
```