Assignment4

JingLiu_18231917

7 March 2019

Introduction

this report analysis the documents clustering by hierarchical Clustering and using diiferent representation ways to visualing and analysis the text of each cluster.

```
#install.packages('tm')
#install.packages('SnowballC')
#install.packages('wordcloud')
#install.packages('RColorBrewer')
#install.packages('dplyr')
#install.packages("cluster")
#install.packages("ggdendro")
#install.packages("HSAUR")
#install.packages("fpc")
#install.packages("skmeans")
#install.packages("plyr")
#install.packages('factoextra')
#install.packages('NbClust')
library(tm)
```

```
## Loading required package: NLP
```

```
library(SnowballC)
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(RColorBrewer)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:NLP':
##
##     annotate
```

```
library(ggdendro)
library(cluster)
library(HSAUR)
```

```
## Loading required package: tools
```

```
library(fpc)
library(plyr)
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
library(factoextra)
```

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

```
library(NbClust)
```

Data Pre-processing

the corpus consists of 7,142 short text documents, firstly pre-process the data, here remove punctuation, then convert all token to lower case, remove stop words, and strip whitespace on documents and stem document and etc.

```
courp1<-VCorpus(DirSource("corpus",encoding = "UTF-8"),readerControl=list(language ="eng"))

toSpace <- content_transformer(function (x , pattern ) gsub(pattern, " ", x))

courp1 <- tm_map(courp1, toSpace, "/")
courp1 <- tm_map(courp1, toSpace, "/.")
courp1 <- tm_map(courp1, toSpace, "@")
courp1 <- tm_map(courp1, toSpace, "\\|")
courp1 <- tm_map(courp1, content_transformer(tolower))
courp1 <- tm_map(courp1, removeWords, stopwords("english"))
courp1 <- tm_map(courp1, removePunctuation)
courp1 <- tm_map(courp1, removeNumbers)
courp1 <- tm_map(courp1, removeWords, c(letters))
courp1 <- tm_map(courp1, stripWhitespace)
courp1 <- tm_map(courp1, stemDocument)
```

DocumentTermMatrix Generation

Using DocumentTermMatrix the row is each documents ans the columnn is each word, using the TF,IDF to represent weights, the sparsity_threshold is used for remove sparse terms, for the threshold 0.95, which can keep the matrixs dense. then using as.amtrix() convert it to matrix,then remove all 0 rows and generate archive.dtm.mat again.

```
# Stem the words in the archive
archive.dtm <- DocumentTermMatrix(courp1, control = list(weighting = function(x) weightTfIdf
(x, normalize = TRUE)))
sparsity_threshold = 0.9
archive.dtm<-removeSparseTerms(archive.dtm, sparsity_threshold)
archive.dtm.matrix <- archive.dtm %>% as.matrix()
archive.dtm.matrix <- archive.dtm.matrix[rowSums(archive.dtm.matrix^2) !=0,]
```

Most frequently Terms

Sort the archive.dtm.matrix by the frequency of each word through sum all rows for same word, then generate the new dataframe which can see the top 10 most frequency words.

```
#call inspect (archive.tdm) if you want to see the new sparsity level

v <- sort(colSums(archive.dtm.matrix),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 10)
```

```
##            word     freq
## window   window 36.15975
## use         use 30.32332
## univers univers 30.09768
## can         can 27.92505
## thank     thank 27.78929
## file       file 27.65457
## drive     drive 27.64914
## will       will 26.51326
## know       know 26.43310
## comput   comput 26.03154
```

World Cloud of whole corpus

Here generate the wordcloud for whole corpus, from the corpus we can see the window is most frequently words in whole corpus.

```
set.seed(2356)
wordcloud(words = d$word, freq = d$freq, min.freq = 2,
          max.words=100, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```

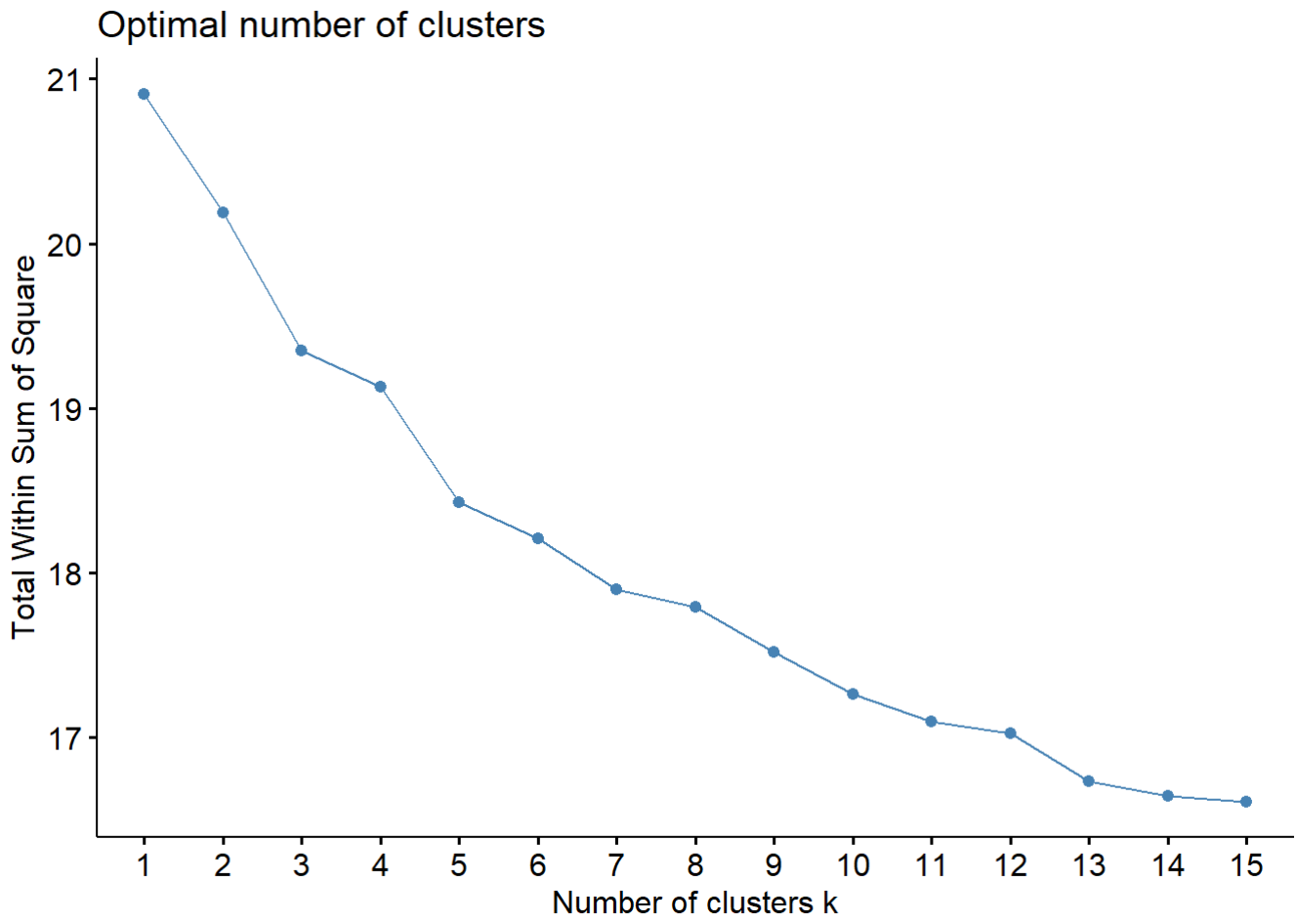
Randomly generate samples for 30% of corpus as dataset.

```
percent = 30
sample_size = nrow(archive.dtm.matrix) * percent/100
set.seed(123)
archive.dtm.mat.sample <- archive.dtm.matrix[sample(1:nrow(archive.dtm.matrix), sample_size,
 replace=FALSE),]
```

Determin K Value

reference:https://blog.csdn.net/harry_128/article/details/80523568 (https://blog.csdn.net/harry_128/article/details/80523568) For the determine k value for clustering, the original code is not works, here I use fviz_nbclust, this function can show a line chart which the best K is the turning point that the line from rapid decline to decline slowly, so here I choose 5 as the k value

```
fviz_nbclust(archive.dtm.mat.sample, kmeans, method="wss",k.max=15)
```



Hierarchical Clustering

After getting the k value and all victors, firstly , we need calculate the distance of each doc, here using distance function, and use cosine function to get the similarity of calculate, then convert it to be a matrix, get distance between max similarity with itself, then generate the distance matrix of each cluster.

```
# philentropy library provides a number of distance/similarity measures
# including cosine which we use for group documents
#install.packages("philentropy")
library(philentropy)
```

```
## Warning: package 'philentropy' was built under R version 3.5.3
```

```
# from philentropy library. Slower than dist function, but handles cosine similarity
sim_matrix<-distance(archive.dtm.mat.sample, method = "cosine")
```

```
## Metric: 'cosine'; comparing: 1035 vectors.
```

Distance Matrix

here generate the distance matrix for the distance of each document, then according to the distance to generate hierarchical clustering

```
# for readiblity (and debugging) put the doc names on the cols and rows
colnames(sim_matrix) <- rownames(archive.dtm.mat.sample)
rownames(sim_matrix) <- rownames(archive.dtm.mat.sample)

# cosine is really a similarity measure (inverse of distance measure)
# we need to create a distance measure for hierarchical clustering
max_sim <- max(sim_matrix)

dist_matrix <- as.dist(max_sim-sim_matrix)

# hierarchical clustering
archive.dtm.sample.dend <- hclust(dist_matrix, method = "ward.D")
```

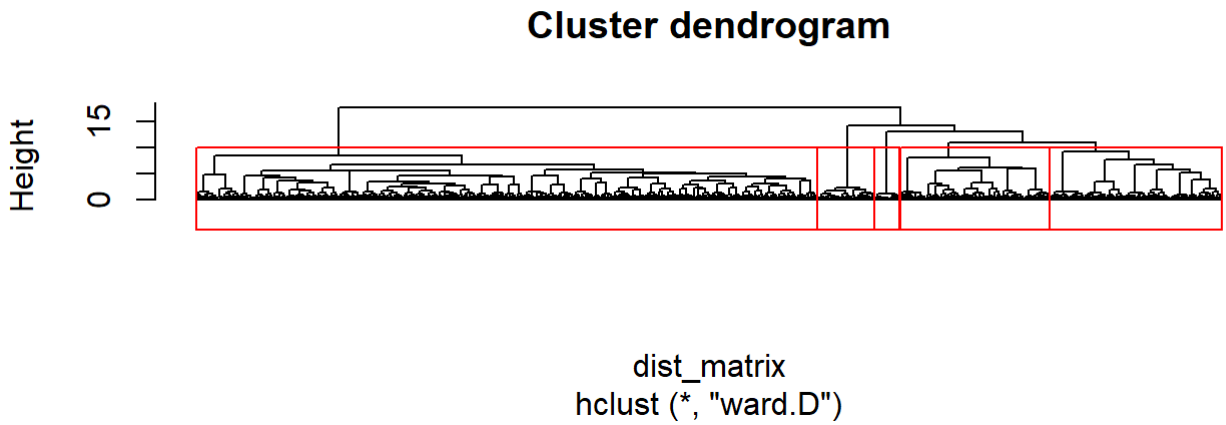
Cluster Dendogram

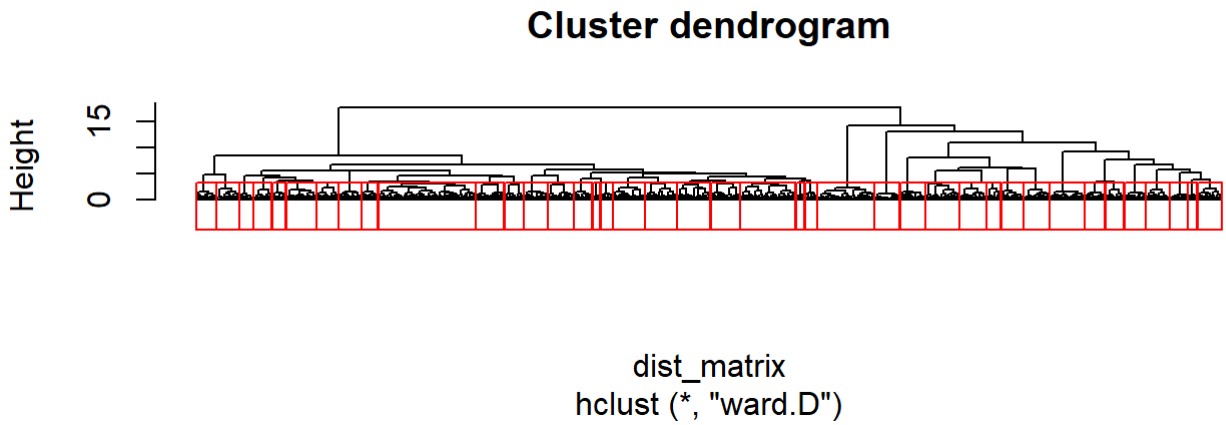
the dendogram shows that the hierarchical relationship between objects,it is commonly created as an output from hierarchical clustering, the height of the dendrogram indicates the distance between the clusters, so from the first cluster dendrogram we can see when k-value=5,here we can see that the cluster 1 has much more documents than others and the cluster 4 and 5 has a short distance but cluster 1 has a long distance with rest of cluters. The second dendogram is represent that give a fix height 3.2,how many clusters will be decide, the height value very low so that there are so many clusters and get a fuzzy relationship between clusters.

```
# plot the dendogram
# we hope to see some structure that reflects the finding of the kmeans algorithm
set.seed(2584)
par(mfrow=c(2,1))
plot(archive.dtm.sample.dend, hang= -1, labels = FALSE,  main = "Cluster dendrogram", sub = N
ULL, xlab = NULL, ylab = "Height")

# here rect.hclust creates  rectangles around the dendogram for k number of clusters
rect.hclust(archive.dtm.sample.dend, k =5, border = "red")
plot(archive.dtm.sample.dend, hang= -1, labels = FALSE,  main = "Cluster dendrogram", sub = N
ULL, xlab = NULL, ylab = "Height")

# here rect.hclust creates a rectangles around the clusters existing at a given height, h
rect.hclust(archive.dtm.sample.dend, h = 3.2, border = "red")
```





Cut Tree here inspect the clustering induced by the dendogram by making a horizontal cut that gives a particular value of k, and then examining the document contents in each cluster

```
#install.packages("gplots")
library(gplots)
```

```
## Warning: package 'gplots' was built under R version 3.5.3
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:wordcloud':
##
##     textplot
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(stats)

# define the cut to make
# you can specify the height at which to cut using argeument h
# or the number of clusters using k
#cut <- max(corpus.dtm.sample.dend$height)/2.1

# number of clusters we wish to examine
k=5

# call the cutree function
# cutree returns a vector of cluster membership
# in the order of the original data rows
archive.dtm.sample.dend.cut <- cutree(archive.dtm.sample.dend, k=k)

#number of clusters at the cut
m <- length(unique(archive.dtm.sample.dend.cut))

# create a data frame from the cut
archive.dtm.sample.dend.cut <- as.data.frame(archive.dtm.sample.dend.cut)

#add a meaningful column namane
colnames(archive.dtm.sample.dend.cut) = c("cluster")

# add the doc names as an explicit column
archive.dtm.sample.dend.cut$docs <- rownames(archive.dtm.sample.dend.cut)

# for the journal datasets each document has a name like 'a1', 'b1 etc
# As before, I remove the number and full stop so that the docs column
# only contains the class name ('a', 'b', etc)

archive.dtm.sample.dend.cut$docs<-lapply(archive.dtm.sample.dend.cut$docs, tm::removeNumbers)
#corpus.dtm.sample.dend.cut$docs<-lapply(corpus.dtm.sample.dend.cut$docs, tm::removePunctuati
on)

# I unlist the list assigned by rownames to $docs
archive.dtm.sample.dend.cut$docs <- unlist(archive.dtm.sample.dend.cut$docs)
```

Confusion Matrix

Here create the frequency table for each clusters, we can see the cluster1 has 627 documents and the cluster 4 and 5 have few documents.

```
# create a frequency table
archive.dtm.sample.dend.cut.table <-table(archive.dtm.sample.dend.cut$cluster, archive.dtm.sa
mple.dend.cut$docs)
#displays the confusion matrix
archive.dtm.sample.dend.cut.table
```

```
##
##     doc
##   1 627
##   2 151
##   3 174
##   4  26
##   5  57
```

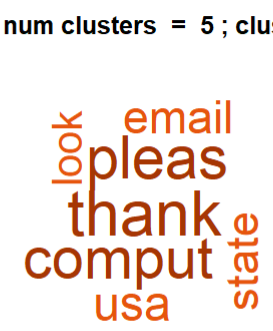
Word Could

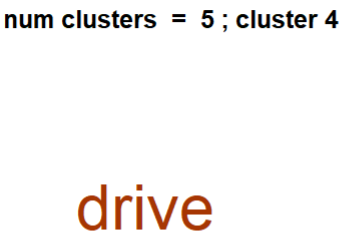
As we see on confusion matrix, for cluster 1, there are lots of documents,the keywords is like article and work, university,write and etc, it is likely belong to a cluster of study, and the second cluster doesn't clear like first one, here we have usa, comput,thanks and please. the third one is very clear, which include file, software, program, this cluster looks like describing a area of computer science. while for the last two clusters, there are only one word on the plot due to few documents, the cluster4 only descibe drive, it may be related with car, the last cluster is about window.

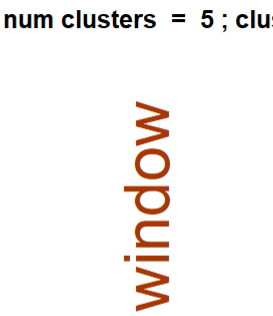
```
#number of clusters at the cut
m <- length(unique(archive.dtm.sample.dend.cut$cluster))

set.seed(1478)
par(mfrow=c(2,3))

# for each cluster plot an explanatory word cloud
for (i in 1:m) {
  #the documents in  cluster i
  cut_doc_ids <-which(archive.dtm.sample.dend.cut$cluster==i)
  #the subset of the matrix with these documents
  archive.dtm.sample.mat.cluster<- archive.dtm.mat.sample[cut_doc_ids, ]
  # sort the terms by frequency for the documents in this cluster
  v <- sort(colSums(archive.dtm.sample.mat.cluster),decreasing=TRUE)
  d <- data.frame(word = names(v),freq=v)
  # call word cloud function
  wordcloud(words = d$word, freq = d$freq, scale=c(3,.2), min.freq = 2,
          max.words=60, random.order=FALSE, rot.per=0.35,
          colors=c('#feedde','#fdbe85','#fd8d3c','#e6550d','#a63603'))
 title(paste("num clusters  = ", k, "; cluster", i))
}
```







Tree Map

This is another way to display the terms of each cluster, it use tree map to display, the different size and colors represent the frequency of terms, using tree map I can see that the most shallow color is represent two most frequently word in cluster 4 and cluster 5 respectively, it's drive and window, then for cluster 3 here is problem,program and file with frequency of 4, then the cluster 2 is thank, please, unir, state,usa, for the clster1,there are not too much distance of frequency of each word, the frequency of each word is lower than 2 or equal to 2.

```
#install.packages('treemapify')
library(treemapify)
```

```
## Warning: package 'treemapify' was built under R version 3.5.3
```

```
#number of clusters at the cut
m <- length(unique(archive.dtm.sample.dend.cut$cluster))

# number of terms per cluster to show
n <-20

#intialise an empty data frame
#fields initiliased with empty vectors
df <- data.frame(word=character(), freq = double(),cluster = integer())
# for each cluster plot an explanatory word cloud


for (i in 1:m) {
  #the documents in  cluster i
  cut_doc_ids <-which(archive.dtm.sample.dend.cut$cluster==i)

  #the subset of the matrix with these documents
  archive.dtm.sample.mat.cluster<- archive.dtm.mat.sample[cut_doc_ids, ]

  # sort the terms by frequency for the documents in this cluster
  v <- sort(colSums(archive.dtm.sample.mat.cluster),decreasing=TRUE)
  d <- data.frame(word = names(v),freq=v, cluster=i)

  # we might want scale so that high frequencies in large cluster don't predominate
  d[,2] <- scale(d[,2],center=FALSE, scale=TRUE)

  # take first n values only
  d <-d[1:n,]

  #bind the data for this cluster to the df data frame created earlier
  df<- rbind(df,d)
}
# the geom_treemap seems to only like vectors of values
df$freq <- as.vector(df$freq)

# simple function to rename the values in the cluster column as "cluster 1, cluster 2, etc"
clust_name<-function(x){
   paste("cluster", x)
}

# apply the function to the 'cluster' column
 df$cluster<- as.character(apply(df["cluster"], MARGIN = 2,FUN =clust_name ))


gg<- ggplot(df, aes(area = freq, fill = freq, subgroup=cluster, label = word)) +
geom_treemap() +
geom_treemap_subgroup_border() +
geom_treemap_subgroup_text(place = "centre", grow = T, alpha = 0.5, colour ="#99d8c9", fontfa
ce = "italic", min.size = 10) +
geom_treemap_text(fontface = "italic", colour = "white", place = "centre", grow = TRUE)
gg
```

Tree Map

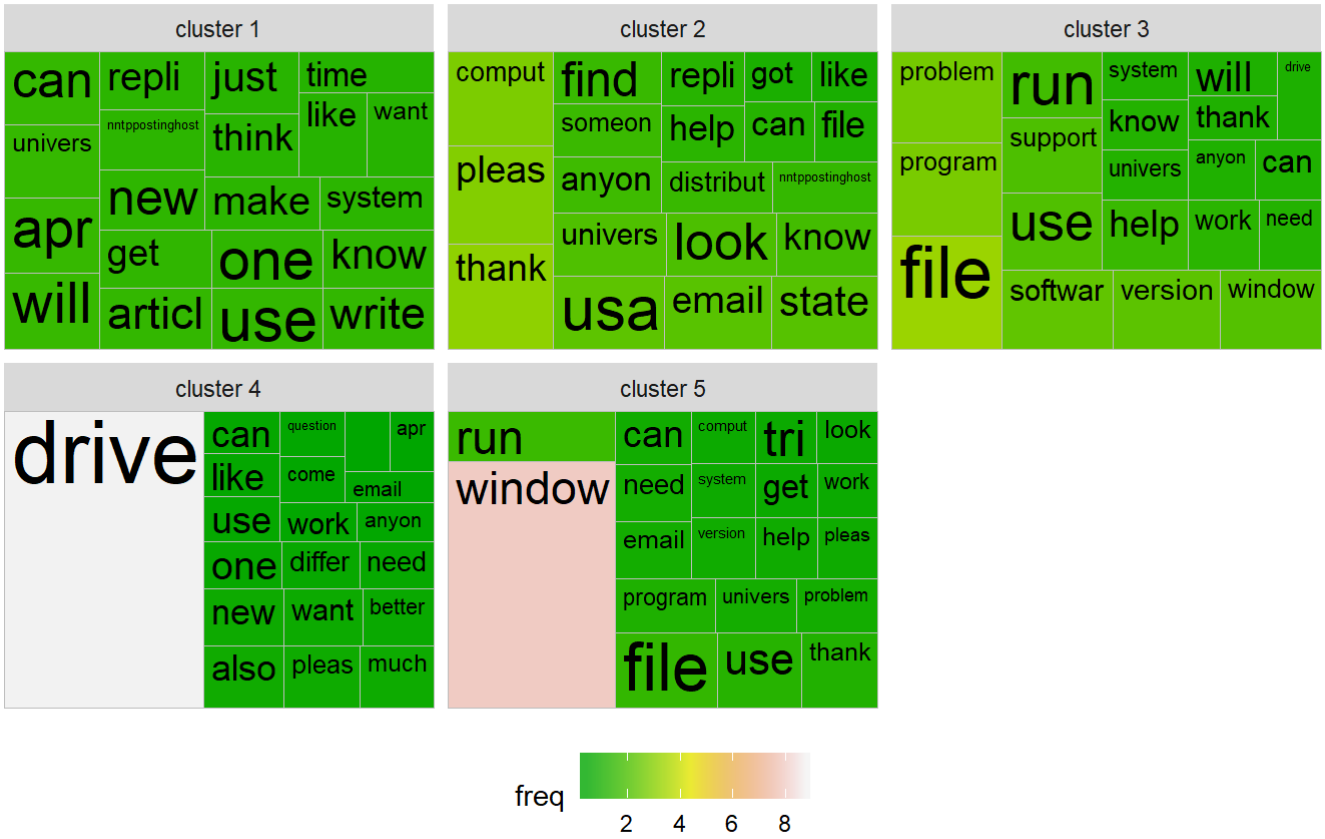
Here is most frequency terms in each cluster, it's obviously that the most frequently term is windom, its represrnt as pink and max size, then for cluster 4 is drive, then for cluster 3 it's file and cluster 2 is compute, please, thank, the cluster1 represent is will, artile and can.

```
gg<- ggplot(df, aes(area = freq, fill = freq, subgroup=cluster, label = word)) +
geom_treemap() +
  geom_treemap_text(grow = T, reflow = T, colour = "black") +
  facet_wrap( ~ cluster) +

  scale_fill_gradientn(colours = terrain.colors(n, alpha = 0.8)) +
  theme(legend.position = "bottom") +
  labs(title = "The Most Frequent Terms in each cluster ", caption = "The area of each term i
s proportional to its relative frequency within cluster")

gg
```

The Most Frequent Terms in each cluster



The area of each term is proportional to its relative frequency within cluster

Conclusion

In conclusion, my report represent the clustering situation and I set 5 clustering according to determing K by fviz_nbclust function, then using cluster dendrogram to diaplay the relation and distance of each cluster, from the confusion table can get the idea of each cluster's document frequency,like cluater 1 have most documents while cluster 4 have few documents, and using wordcloud and tree map to get the most frequently terms in each cluster so that get the the main content of each cluster represent.