Project 3

Title: Text Analytics in R

Group Member: Shijie Lu, RunyuPan, Rongrui Jiang, Chutian Chen, Yuxi Liu

Our Steps:

1. Data Set: acq:

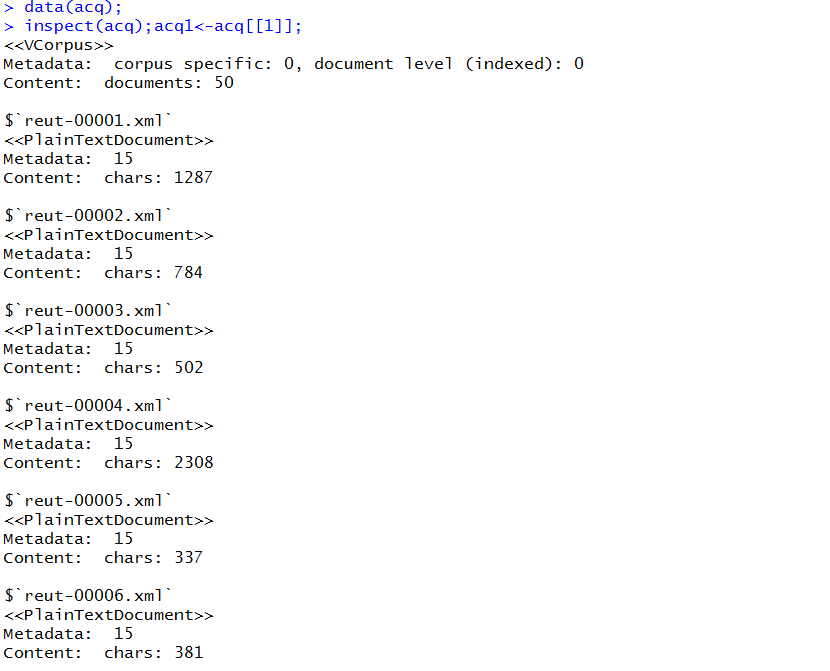
Firstly, we install the tm package to use the function acq. import the library tm, then start to try the functions in lecture 8.

Step a. Try the functions in lecture 8. What happens? Does it yield anything understandable about the documents.

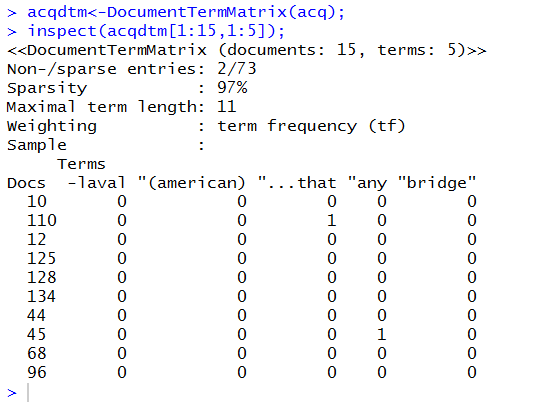
Also answer to the questions: Show a listing of all R functions that you have written

Presentation and discussion of results from the experiments that you run using the different functions from Lecture 7: parts (a) through (h) 2 points each. Include plots where applicable.

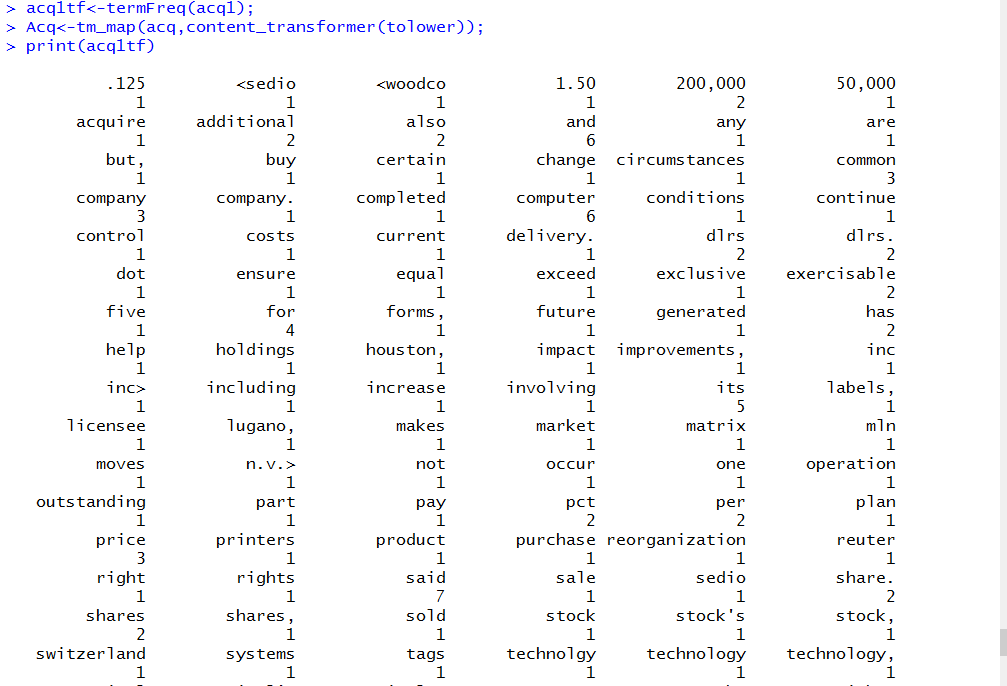
In order to inspect the dataset, we use the function inspect(acq) to access the information.



By using the function DocumentTermMatrix, we change it into matrix form. Then we inspect the specific data sets with inspect function to output the outcome.



In order to obtain the word frequency, the function termFreq() is used to check the frequency of all the words. The outcome is printed below.



Then we start to clean the documents. Transformations are done via the tm\_map() function which applies (maps) a function to all elements of the corpus.

Acq<-tm\_map(acq,content\_transformer(tolower));

removeNumPunct <- function(x) gsub("[^[:alpha:][:space:]]\*", "", x);

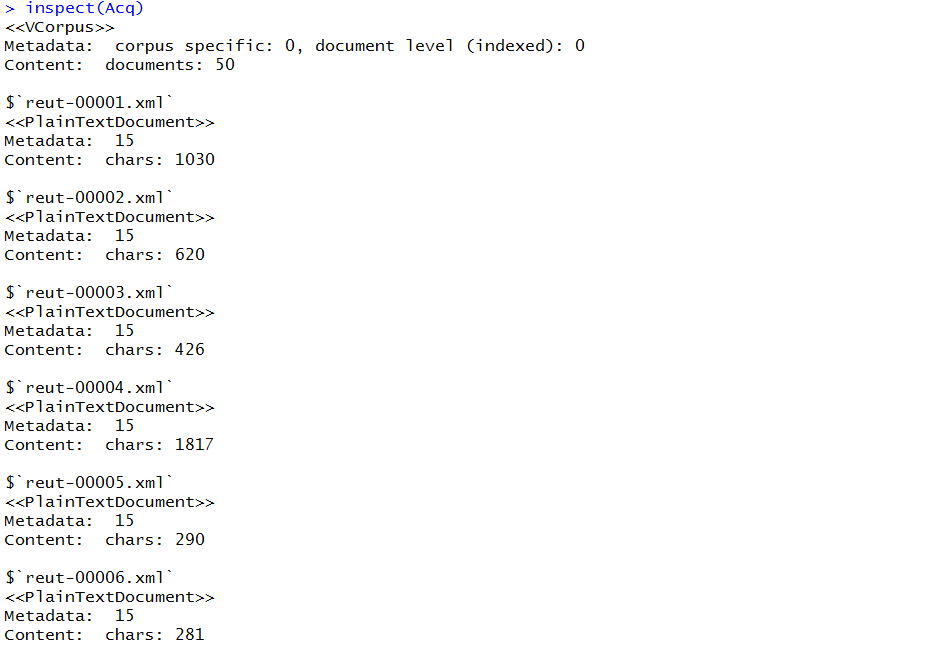
removeBackslashN<-content\_transformer(function(x, pattern)gsub(pattern, " ", x));

Acq <- tm\_map(Acq, content\_transformer(removeNumPunct));

Acq <- tm\_map(Acq, removeBackslashN, "\n");

Acq <- tm\_map(Acq, removeWords, stopwords('english'))

Those functions are used to remove some words and punctuation like \n. Then we check the data, we find the information have changed.



Step b. Find the 15 longest documents.

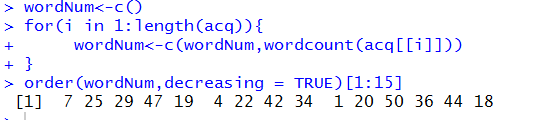
for(i in 1:length(acq)){

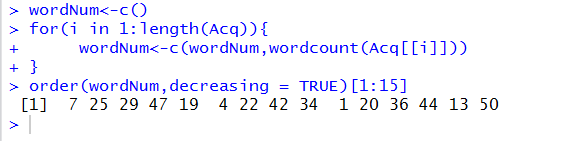
wordNum<-c(wordNum,wordcount(acq[[i]]))

}

order(wordNum,decreasing = TRUE)[1:15]

We use a loop to check the longest documents every time and sort them by decreasing.





We do the sorting process both before cleaning and after cleaning the documents.

Step c: For each document work through the examples given in Lecture 7 to display the dendrogram and the WordCloud.

Acqtdm<-TermDocumentMatrix(Acq,control=list(wordLengths=c(1,Inf)))

tdm1<- TermDocumentMatrix(Acq[1])

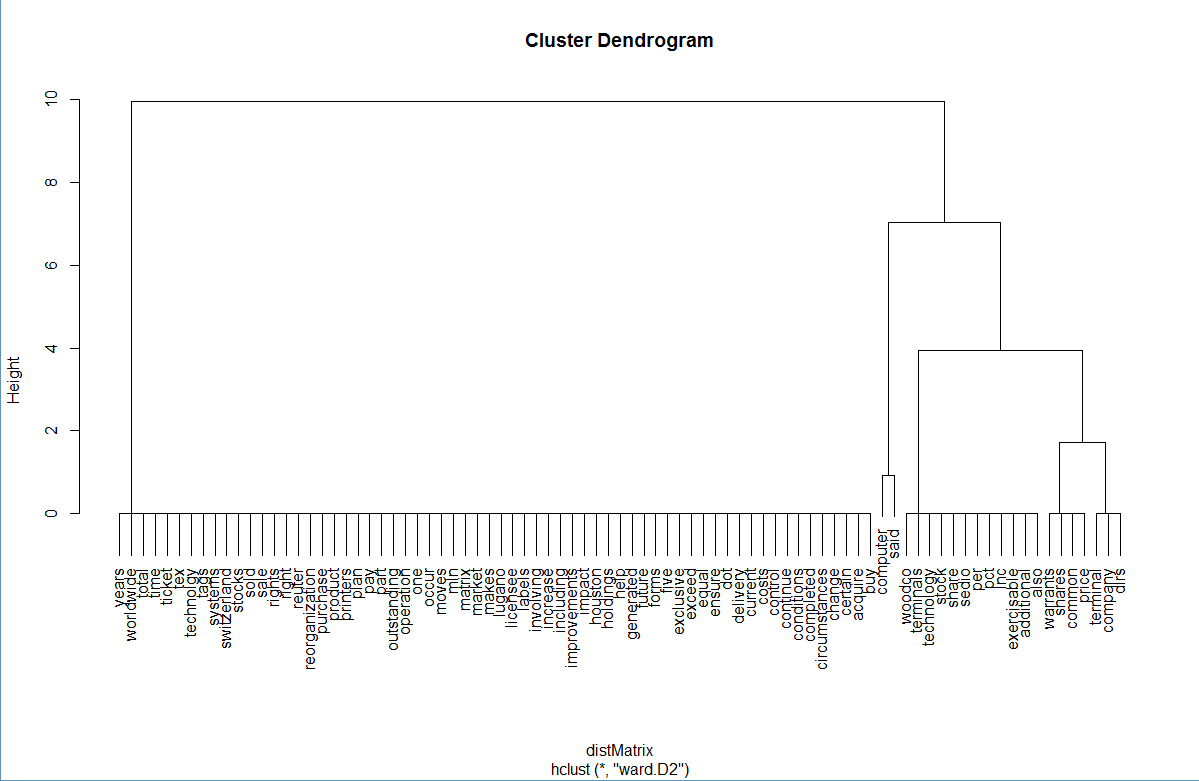
distMatrix<-dist(scale(tdm1))

fit<-hclust(distMatrix,method="ward.D2")

plot(fit)

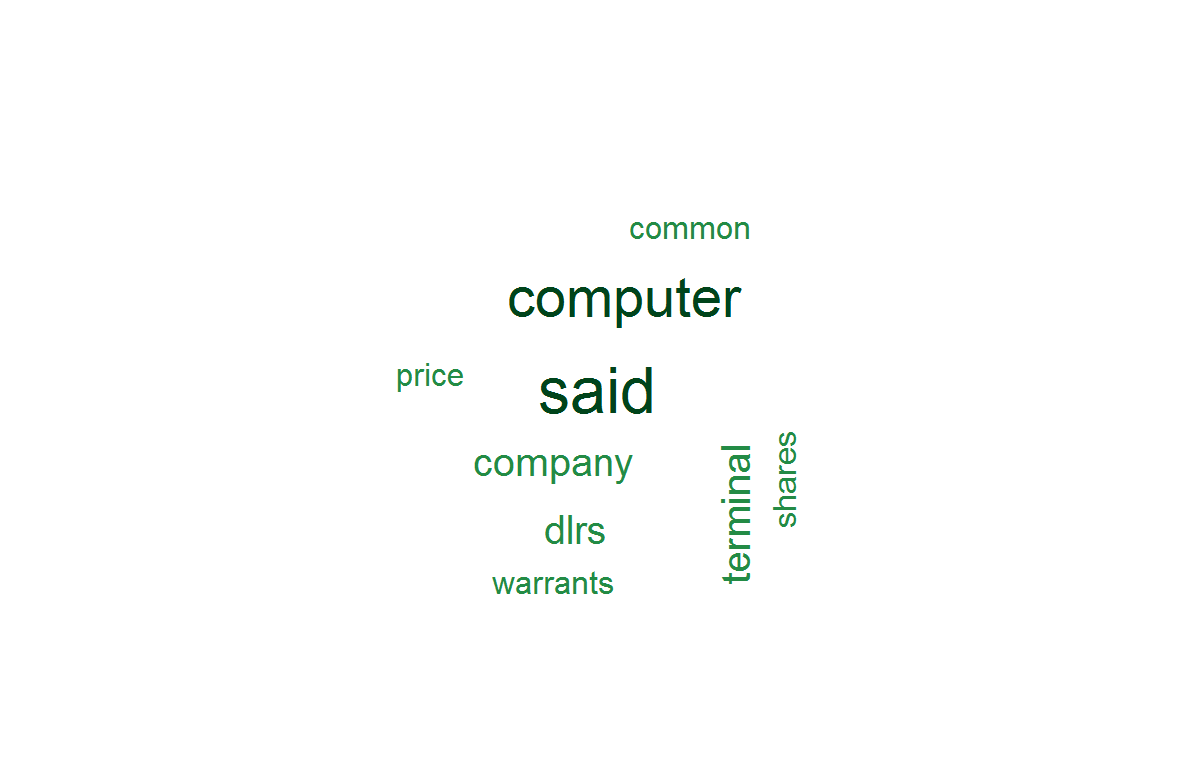
graphics.off()

We use hclust function to produce a cluster dendrogram.



As is depicted, words are sorted by the frequency.

As for the wordcloud, we utilize the sort() function to find the word frequency, and wordcloud() function to process the word cloud plot.



The biggest one in the center is the most frequent used words. Words which are near the center and bigger are more frequently used than those words which are distant from the center and smaller.

Step d. Prior to removing the punctuation, find the longest word and longest sentence in each document from the 15 largest documents.

The function unlist simplifies a list structure x to produce a vector which contains all the atomic components which occur in x. The function lapply returns a list of the same length as x. We use them to find the longest word.

content1<-Acq[[22]]$content

vector<-unlist(strsplit(content1,split="[ ][ ]\*"))

vectorLen<-lapply(vector,nchar)

maxIndex<-which.max(vectorLen)

maxWord<-vector[maxIndex]

The outcome is :



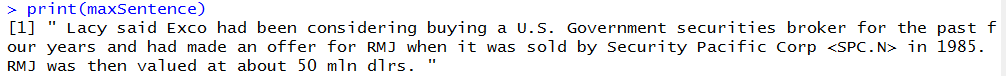
As for the longest sentence of most chars, we use a loop to extract the longest one from all the data.

for( i in 1: length(vector2)){

vector2[i]<-gsub("\n"," ",vector2[i])

}

The outcome is:



Step e: And for the longest sentence of most words, we select the longest one which contains no punctuation. We should remove punctuation in every sentence at first.

for( i in 1: length(vector2)){

vector3[i]<-gsub("[^[:alpha:][:space:]]\*","",vector2[i])

}

vectorlen3<-c()

for(i in 1:length(vector3)){

vectorlen3<-c(vectorlen3, wordcount(vector3[i]))

}

maxIndex3<-which.max(vectorlen3)

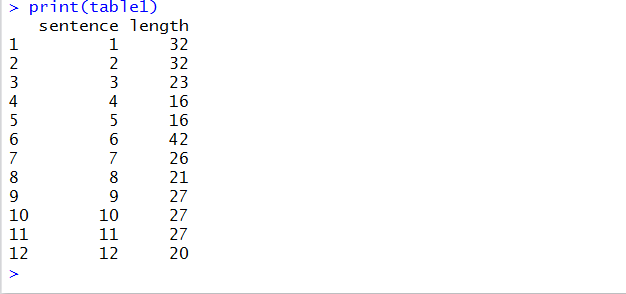
maxSentence2<-vector2[maxIndex3]

Step f: Answer to the question: Print a table of the length of each sentence in each of the 10 documents.

table1<-data.frame(sentence=1:length(vector2), length=vectorlen3)

By attaching data frame, we display the table with the specific length and specific sentence.

The outcome is:



Answer to the question: For each sentence of each document, remove the punctuation. Display the sentences.

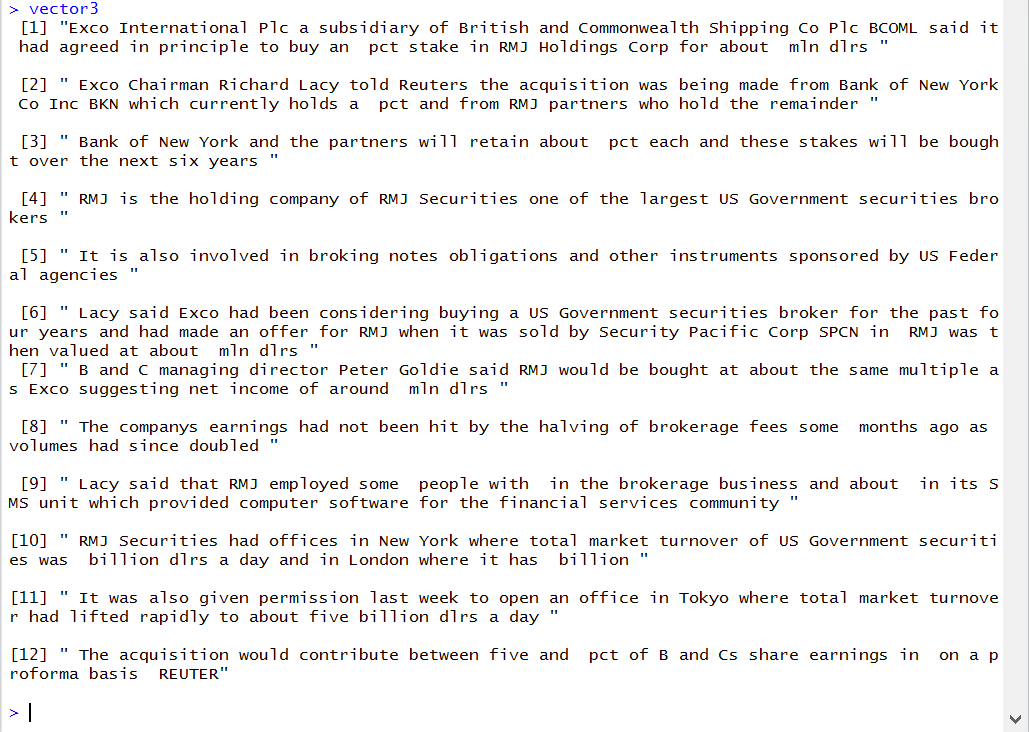
To remove the punctuation, we use the gsub() fuction and a loop on every sentence.

for( i in 1: length(vector2)){

vector3[i]<-gsub("[^[:alpha:][:space:]]\*","",vector2[i])

}

After removing the punctuation, the outcome is:



Answer to the question: Write an R function to search through the documents to find a specific word or phrase. Print the document number, line number, and word index in the sentence. Demonstrate with three examples. Use words of 6 characters or more as your test cases. 3 points.

We give these R functions to search words and three examples:

key<-"computer"

ACQ<-tm\_map(acq,content\_transformer(tolower));

ACQ <- tm\_map(ACQ, removeBackslashN, "\n");

ACQ <- tm\_map(ACQ, removePunctuation);

for(i in 1:50){

if(str\_detect(ACQ[[i]]$content,key)){

print("this word/phrase is shown in document: ")

print(i)

vectortemp<-ACQ[[i]]$content

# deal with some sentences starting at space

vector4<-unlist(strsplit(vectortemp, split=" "))

for(j in 1: length(vector4)){

if(vector4[j]!=""&&str\_detect(vector4[j],key)){

print("this word is shown in the sentence: ")

print(j)

sentence<-gsub(" ","", vector4[j])

vector5<-unlist(strsplit(sentence,split=" "))

for(k in 1:length(vector5)){

if(vector5[k]!=""&&str\_detect(vector5[k],key)){

print("index: ")

print(k)

}

}

}

}

}

}

The output for “computer”:

[1] "this word/phrase is shown in document: "

[1] 1

[1] "this word is shown in the sentence: "

[1] 1

[1] "index: "

[1] 1

[1] "this word is shown in the sentence: "

[1] 3

[1] "index: "

[1] 1

[1] "index: "

[1] 24

[1] "this word is shown in the sentence: "

[1] 5

[1] "index: "

[1] 1

[1] "this word is shown in the sentence: "

[1] 7

[1] "index: "

[1] 1

[1] "index: "

[1] 4

[1] "this word/phrase is shown in document: "

[1] 22

[1] "this word is shown in the sentence: "

[1] 9

[1] "index: "

[1] 24

[1] "this word/phrase is shown in document: "

[1] 24

[1] "this word is shown in the sentence: "

[1] 1

[1] "index: "

[1] 26

The output for“world”:

[1] "this word/phrase is shown in document: "

[1] 1

[1] "this word is shown in the sentence: "

[1] 5

[1] "index: "

[1] 39

[1] "this word/phrase is shown in document: "

[1] 25

[1] "this word is shown in the sentence: "

[1] 2

[1] "index: "

[1] 21

[1] "this word/phrase is shown in document: "

[1] 36

[1] "this word is shown in the sentence: "

[1] 5

[1] "index: "

[1] 17

The output for “machine”:

[1] "this word/phrase is shown in document: "

[1] 45

[1] "this word is shown in the sentence: "

[1] 1

[1] "index: "

[1] 20

Step g. For each word print its part of speech using the Wordnet package.

We have shown the output in the file “nice.txt” in the zip package.

Step h. Analyze word frequency using functions from package zipfR.

# draw frequency cloud

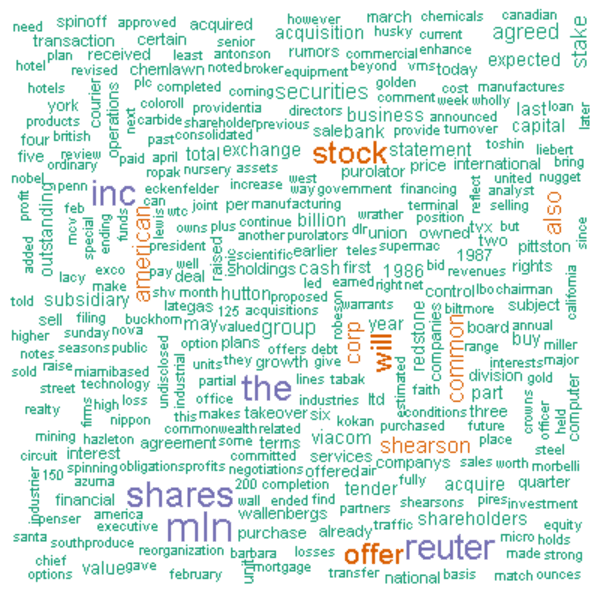
acqdtm=as.matrix(DocumentTermMatrix(acq));

fcnt=sort(colSums(acqdtm),decreasing = TRUE);

wcl=data.frame(word=names(fcnt),freq=fcnt)

wordcloud(wcl$word,wcl$freq,randmom.order=FALSE,colors=brewer.pal(8,"Dark2"))

With these Rfunctions, we can see a frequency cloud like this



Answer to the question: Analysis of what this project helped you learn about data science, e.g., the exploration of data which is what you have been doing: 3 points

From the project above, we have learned a systematic analysis for an interesting dataset. Firstly, we install packages and try the functions in lecture 8. Then clean the documents and find the 15 longest documents before and after cleaning. After that, we display the dendrogram and wordcloud. Thirdly, we find the longest word and longest sentences, which contain most chars or words. Then we remove punctuation in every sentence and display them. At last we print the speech of each word and analysis the word frequency. From the whole workflow, we have learned an advanced way do analysis for the dataset, we have overcome many problems, the cooperation among our team members have been strengthened a lot.