Milestone Porject - Coursera

Sora

June 10th, 2016

This report is a milestone report for the Coursera course, which aims to build a model for natural language process.

There are three parts in this report. The first part is to have a general idea about the data we are about to process. This includes knowing the storage requirement of loading data, and getting to know what the data looks like, etc. The second part is to clean the data. Becuse corpus contains numbers, capitals, punctuations, whitespace, we will try to eliminate those, create a clean data set. The third part starts to get into the data, calculating the work frequencies to help the prediction in the upcoming procedures.

Note: the fist element in list is en_US.blog,txt, the 2nd is en_US.news.txt, the 3rd element is en_US.twitter.txt

Part 1: First Glance at Data

Following is some basic information and file size. (Originally I was using lapply to do this, but for some unknown reason R markdown did not accpet lapply function)

```r
library(R.utils)
library(gdata)
#for blog txt
file.info(a[1])
```

```
##             size isdir mode               mtime               ctime
## check_cache  102  TRUE  755 2016-06-11 15:16:49 2016-06-11 15:16:49
##                           atime uid gid uname grname
## check_cache 2016-06-11 15:16:49 501  20  bean  staff
```

```r
file.info(a[2])
```

```
##           size isdir mode               mtime               ctime
## check.Rmd 5489 FALSE  644 2016-06-11 15:16:48 2016-06-11 15:16:48
##                         atime uid gid uname grname
## check.Rmd 2016-06-11 15:16:49 501  20  bean  staff
```

```r
file.info(a[3])
```

```
##                      size isdir mode               mtime
## en_US.blogs.txt 210160014 FALSE  644 2014-07-22 10:13:06
##                               ctime               atime uid gid uname
## en_US.blogs.txt 2016-05-30 16:58:13 2016-06-11 15:16:49 501  20  bean
##                 grname
## en_US.blogs.txt  staff
```

```r
humanReadable(file.size(a[1]),standard = "SI", unit = "MB")
```

```
## [1] "1e-04 MB"
```

```r
#for news txt
humanReadable(file.size(a[2]),standard = "SI", unit = "MB")
```

```
## [1] "0.005 MB"
```

```r
#for twitter txt
humanReadable(file.size(a[3]),standard = "SI", unit = "MB")
```

```
## [1] "210.2 MB"
```

Next, we count the lines of each file.

```r
countLines("en_US.twitter.txt") #2360148
```

```
## [1] 2360148
## attr(,"lastLineHasNewline")
## [1] TRUE
```

```r
countLines("en_US.news.txt")  #1010242
```

```
## [1] 1010242
## attr(,"lastLineHasNewline")
## [1] TRUE
```

```r
countLines("en_US.blogs.txt") #899288
```

```
## [1] 899288
## attr(,"lastLineHasNewline")
## [1] TRUE
```

Next, we count the total number of words of each file.

Part 2: Clean Data

There are two steps in part. First, base on the total number of lines in each text file, we will randomly select some lines and read them, this is our training set.

Second, we will clean the training set.

```r
# do result=line(file) to randonly select the readling lines in three files
line=function(x)
{
  a=c(2360148,1010242,899288)
  select=list()
  for(i in 1:length(a))
  {
    choice=rbinom(a[i],1,0.02)
  num=c(1:a[i])
  cho=choice*num
  cho=cho[cho!=0]
  select[[i]]=cho
  }
  return (select)  #the output is the index oflines to be read in each file
}
file=c("en_US.blogs.txt","en_US.news.txt","en_US.twitter.txt")
result=line(file)
#result[[1]] = blog; result[[2]]=news; result[[3]]=twitter

#now, read the lines according to index
one=result[[1]]
two=result[[2]]
three=result[[3]]
read=list()
READING = list()
for(i in 1:(length(one) - 1)) {
  READING[[i]] = scan(file[1], skip = diff(one)[i] - 1, nmax = 1, sep = "\n", what = "raw")
}
read[[1]] = do.call(rbind, READING)

for(i in 1:(length(two) - 1)) {
  READING[[i]] = scan(file[1], skip = diff(two)[i] - 1, nmax = 1, sep = "\n", what = "raw")
}
read[[2]] = do.call(rbind, READING)

for(i in 1:(length(three) - 1)) {
  READING[[i]] = scan(file[1], skip = diff(three)[i] - 1, nmax = 1, sep = "\n", what = "raw")
}
read[[3]] = do.call(rbind, READING)
```

Now we have randomly selected text lines and loaded them. Next, we will clean the training set.

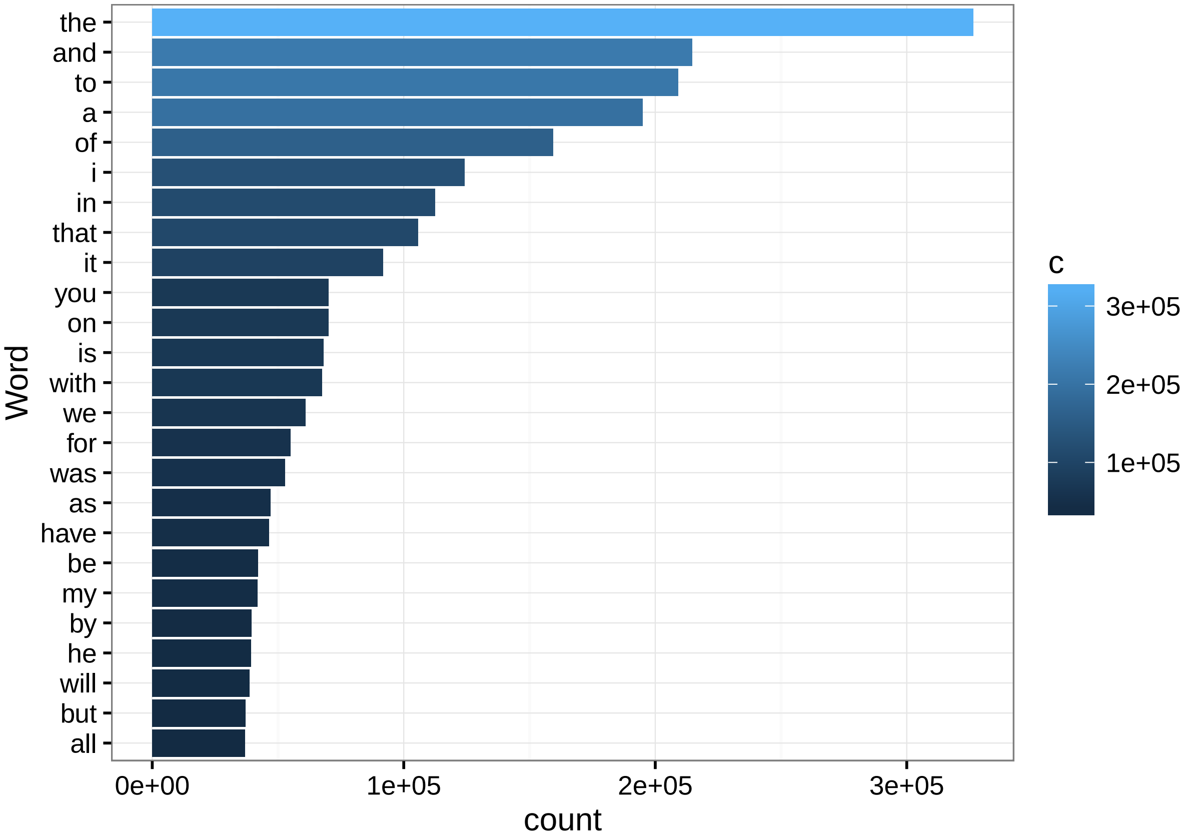
```
#removePunctuation
library(tm)
for(i in 1:length(read))
{
  read[[i]]=removePunctuation(read[[i]])
  read[[i]]=tolower(read[[i]])
  read[[i]]=removeNumbers(read[[i]])
  #read[[i]]=removeWords(read[[i]])
  #after those removing, will create blank space at the removal place
#collapse extra white space into single blank. this shld be the last step
  read[[i]]=stripWhitespace(read[[i]])
}

#transform to matrix
space=matrix(unlist(read),ncol=1,byrow=TRUE)
```

Part 3: Word Frequencies

In this part, we will see which unigram/bigram/trigram words are the most used.

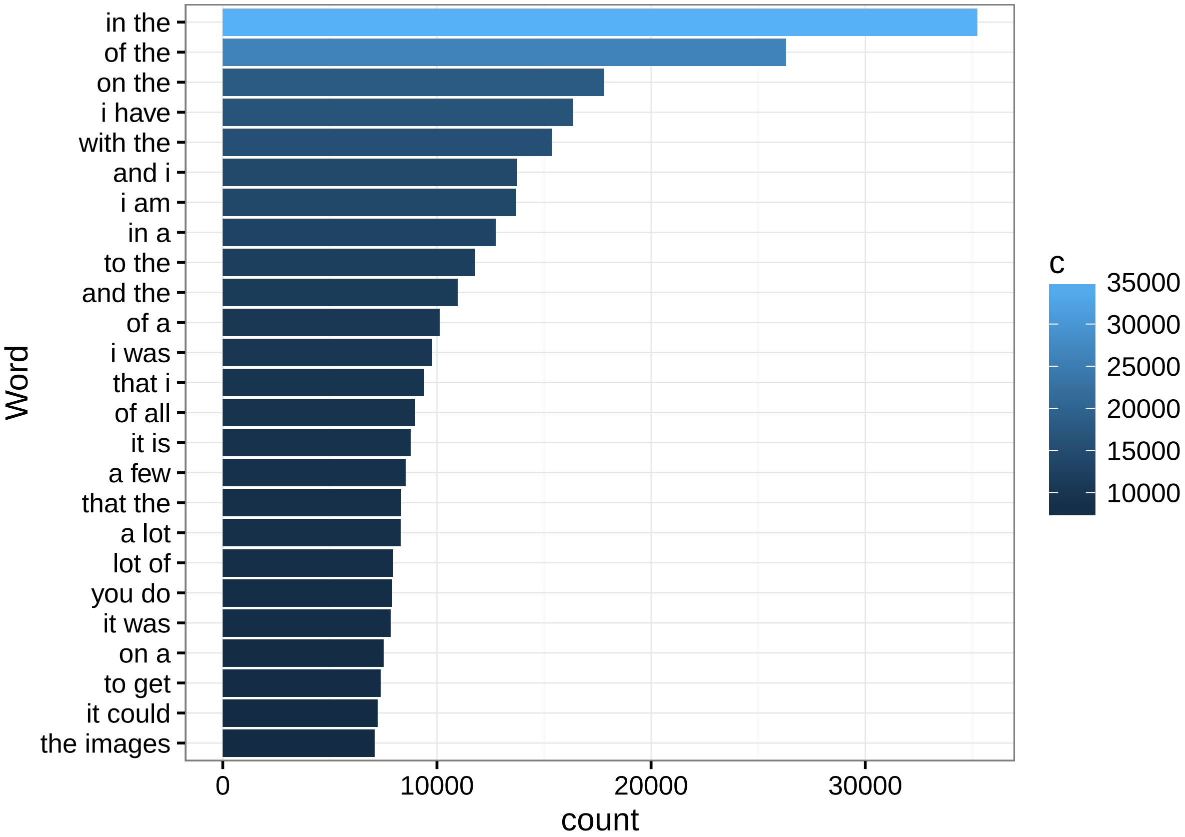
Note: the code for graphing is referenced from http://rstudio-pubs-static.s3.amazonaws.com/41643_9dcfa51dda1d4c9cbe87ca8ed1d3b0a7.html

```
#count the frequency of unigram
library(stats)
b=unlist(strsplit(space, " "))
c=as.data.frame(sort(table(b),decreasing = T))
c$name=row.names(c)
c=c[1:25,]
colnames(c)=c("c","name")
#plot the top 25 frequent unigram words
#use geom_bar instead of geom_historgram.
#geom_histogram is for continuous data (it will do binning) and geom_bar is for discrete data.

#create a function so I don't need to plot 3 times. make sure the col names are c("c","name")
library(ggplot2)
plotgraph=function(x)
  {ggplot(data = x, aes(x = reorder(name, c), y = c)) +
  geom_bar(aes(fill = c), stat = "identity") +
  coord_flip() +
  labs(x = "Word",y="count") +
  theme_bw()
}

plotgraph(c)
```



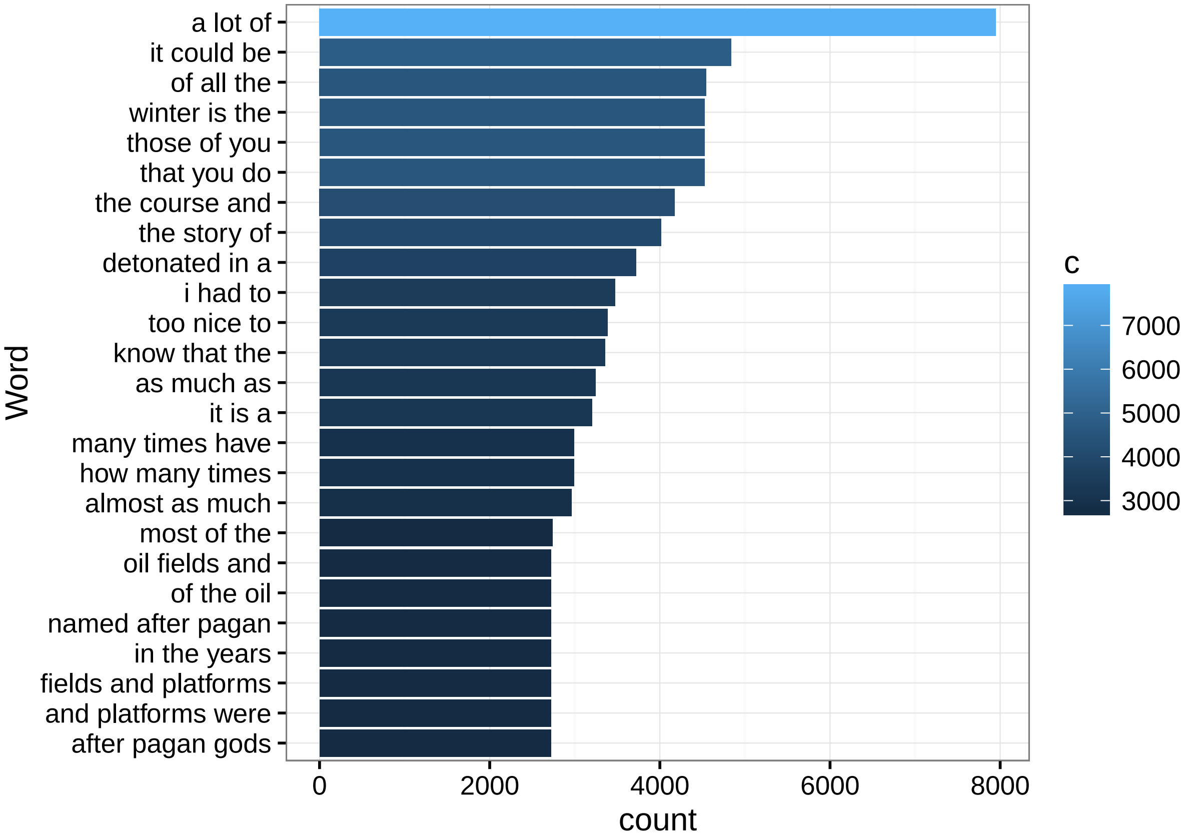
Next, the bigram word frequency.

```
#bigram
#transform data from data frame to textdocument "corpus"
bigram=vapply(ngrams(b,2L),paste,"",collapse = " ")
bigram=as.data.frame(sort(table(bigram),decreasing=T))
bigram$name=row.names(bigram)
row.names(bigram)=c(1:nrow(bigram))
colnames(bigram)=c("c","name")
top25=bigram[1:25,]
plotgraph(top25)
```



Last, the trigram word frequency.

```
#trigram
trigram=vapply(ngrams(b,3L),paste,"",collapse = " ")
trigram=as.data.frame(sort(table(trigram),decreasing=T))
trigram$name=row.names(trigram)
row.names(trigram)=c(1:nrow(trigram))
colnames(trigram)=c("c","name")
top25=trigram[1:25,]
plotgraph(top25)
```


a lot of
it could be
of all the
winter is the
those of you
that you do
the course and
the story of
detonated in a
i had to
too nice to
know that the
as much as
it is a
many times have
how many times
almost as much
most of the
oil fields and
of the oil
named after pagan
in the years
fields and platforms
and platforms were
after pagan gods
Word
0
2000
4000
6000
8000
count
c
7000
6000
5000
4000
3000