170801_coursera_data science capstone(week2)
Prabhakar Brahmandam
14th Jan, 2019

Load data

## Load data

```{r load data}
setwd("c:/Users/prabhakar.brahmandam/Desktop/Caps-Assignment 1/coursera-data-science-capstone-master/data/final/en_US")
blogs<-readLines("en_US.blogs.txt",warn=FALSE,encoding="UTF-8")
news<-readLines("en_US.news.txt",warn=FALSE,encoding="UTF-8")
twitter<-readLines("en_US.twitter.txt",warn=FALSE,encoding="UTF-8")
```

I set the directory and load 3 data.

## Summarize data

```{r summarize}
size_blogs<-file.size(path="c:/Users/prabhakar.brahmandam/Desktop/Caps-Assignment 1/coursera-data-science-capstone-master/data/final/en_US/en_US.blogs.txt")/2^20
size_news<-file.size(path="c:/Users/prabhakar.brahmandam/Desktop/Caps-Assignment 1/coursera-data-science-capstone-master/data/final/en_US/en_US.news.txt")/2^20
size_twitter<-file.size(path="c:/Users/prabhakar.brahmandam/Desktop/Caps-Assignment 1/coursera-data-science-capstone-master/data/final/en_US/en_US.twitter.txt")/2^20
len_blogs<-length(blogs)
len_news<-length(news)
len_twitter<-length(twitter)
nchar_blogs<-sum(nchar(blogs))
nchar_news<-sum(nchar(news))
nchar_twitter<-sum(nchar(twitter))
library(stringi)
nword_blogs<-stri_stats_latex(blogs)[4]
nword_news<-stri_stats_latex(news)[4]
nword_twitter<-stri_stats_latex(twitter)[4]
table<-data.frame("File Name"=c("Blogs","News","Twitter"),
            "File Size(MB)"=c(size_blogs,size_news,size_twitter),
            "Num of rows"=c(len_blogs,len_news,len_twitter),
            "Num of character"=c(nchar_blogs,nchar_news,nchar_twitter),
            "Num of words"=c(nword_blogs,nword_news,nword_twitter))
table
```

Summarize the contents, which has file size, number of rows, number of character and number of words in each file. And make the table

## Clean data

~~~
```{r clean data}
set.seed(12345)
~~~

~~~~
blogs1<-iconv(blogs,"latin1","ASCII",sub="")
news1<-iconv(news,"latin1","ASCII",sub="")
twitter1<-iconv(twitter,"latin1","ASCII",sub="")
rm(blogs)
rm(news)
rm(twitter)
# sample data set only 1% of each file
sample_data<-c(sample(blogs1,length(blogs1)*0.01),
        sample(news1,length(news1)*0.01),
        sample(twitter1,length(twitter1)*0.01))
rm(blogs1)
rm(news1)
rm(twitter1)
```

Data sets are really big, so using sample() function, I sample 1% of each file.

## Build corpus
```{r build corpus}
library(tm)
library(NLP)
corpus<-VCorpus(VectorSource(sample_data))
corpus1<-tm_map(corpus,removePunctuation)
corpus2<-tm_map(corpus1,stripWhitespace)
corpus3<-tm_map(corpus2,tolower)
corpus4<-tm_map(corpus3,removeNumbers)
corpus5<-tm_map(corpus4,PlainTextDocument)
corpus6<-tm_map(corpus5,removeWords,stopwords("english"))
corpus_result<-data.frame(text=unlist(sapply(corpus6,'[',"content")),stringsAsFactors = FALSE)
head(corpus_result)
rm(corpus)
rm(corpus1)
rm(corpus2)
rm(corpus3)
rm(corpus4)
rm(corpus5)
```

Build corpus, and check it making data frame.

## Build N-gram

```{r build N-gram}
library(RWeka)
one<-function(x) NGramTokenizer(x,Weka_control(min=1,max=1))
two<-function(x) NGramTokenizer(x,Weka_control(min=2,max=2))
thr<-function(x) NGramTokenizer(x,Weka_control(min=3,max=3))
one_table<-TermDocumentMatrix(corpus6,control=list(tokenize=one))
two_table<-TermDocumentMatrix(corpus6,control=list(tokenize=two))
thr_table<-TermDocumentMatrix(corpus6,control=list(tokenize=thr))
one_corpus<-findFreqTerms(one_table,lowfreq=1000)
two_corpus<-findFreqTerms(two_table,lowfreq=80)
thr_corpus<-findFreqTerms(thr_table,lowfreq=10)
~~~~

~~~
one_corpus_num<-rowSums(as.matrix(one_table[one_corpus,]))
one_corpus_table<-data.frame(Word=names(one_corpus_num),frequency=one_corpus_num)
one_corpus_sort<-one_corpus_table[order(-one_corpus_table$frequency),]
head(one_corpus_sort)
two_corpus_num<-rowSums(as.matrix(two_table[two_corpus,]))
two_corpus_table<-data.frame(Word=names(two_corpus_num),frequency=two_corpus_num)
two_corpus_sort<-two_corpus_table[order(-two_corpus_table$frequency),]
head(two_corpus_sort)
thr_corpus_num<-rowSums(as.matrix(thr_table[thr_corpus,]))
thr_corpus_table<-data.frame(Word=names(thr_corpus_num),frequency=thr_corpus_num)
thr_corpus_sort<-thr_corpus_table[order(-thr_corpus_table$frequency),]
head(thr_corpus_sort)
```
~~~

Extract the word and frequency of N-grams.

## Plot graph

~~~
```{r plot}
library(ggplot2)
one_g<-ggplot(one_corpus_sort[1:10,],aes(x=reorder(Word,-frequency),y=frequency,fill=frequency))
one_g<-one_g+geom_bar(stat="identity")
one_g<-one_g+labs(title="Unigrams",x="Words",y="Frequency")
one_g<-one_g+theme(axis.text.x=element_text(angle=90))
one_g
two_g<-ggplot(two_corpus_sort[1:10,],aes(x=reorder(Word,-frequency),y=frequency,fill=frequency))
two_g<-two_g+geom_bar(stat="identity")
two_g<-two_g+labs(title="Bigrams",x="Words",y="Frequency")
two_g<-two_g+theme(axis.text.x=element_text(angle=90))
two_g
thr_g<-ggplot(thr_corpus_sort[1:10,],aes(x=reorder(Word,-frequency),y=frequency,fill=frequency))
thr_g<-thr_g+geom_bar(stat="identity")
thr_g<-thr_g+labs(title="Trigrams",x="Words",y="Frequency")
thr_g<-thr_g+theme(axis.text.x=element_text(angle=90))
thr_g
```
~~~

### Clean data

```
set.seed(12345)

blogs1<-iconv(blogs,"latin1","ASCII",sub="")
news1<-iconv(news,"latin1","ASCII",sub="")
twitter1<-iconv(twitter,"latin1","ASCII",sub="")

rm(blogs)
rm(news)
rm(twitter)

# sample data set only 1% of each file
sample_data<-c(sample(blogs1,length(blogs1)*0.01),
               sample(news1,length(news1)*0.01),
               sample(twitter1,length(twitter1)*0.01))

rm(blogs1)
rm(news1)
rm(twitter1)
```

Data sets are really big, so using sample() function, I sample 1% of each file.

### Build corpus

```
library(tm)
```

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

```
library(NLP)

corpus<-VCorpus(VectorSource(sample_data))
corpus1<-tm_map(corpus,removePunctuation)
corpus2<-tm_map(corpus1,stripWhitespace)
corpus3<-tm_map(corpus2,tolower)
corpus4<-tm_map(corpus3,removeNumbers)
corpus5<-tm_map(corpus4,PlainTextDocument)
corpus6<-tm_map(corpus5,removeWords,stopwords("english"))

corpus_result<-data.frame(text=unlist(sapply(corpus6,'[',"content")),stringsAsFactors = FALSE)
head(corpus_result)
```

```
##
## 1
## 2
## 3 ill take  opportunity  diverge   usual take three path  instead  focusing  one last role offer   a
## 4
## 5
## 6
```

```
rm(corpus)
rm(corpus1)
rm(corpus2)
rm(corpus3)
rm(corpus4)
rm(corpus5)
```

Build corpus, and check it making data frame.

### Build N-gram

```
library(RWeka)

one<-function(x) NGramTokenizer(x,Weka_control(min=1,max=1))
two<-function(x) NGramTokenizer(x,Weka_control(min=2,max=2))
thr<-function(x) NGramTokenizer(x,Weka_control(min=3,max=3))

one_table<-TermDocumentMatrix(corpus6,control=list(tokenize=one))
two_table<-TermDocumentMatrix(corpus6,control=list(tokenize=two))
thr_table<-TermDocumentMatrix(corpus6,control=list(tokenize=thr))

one_corpus<-findFreqTerms(one_table,lowfreq=1000)
two_corpus<-findFreqTerms(two_table,lowfreq=80)
thr_corpus<-findFreqTerms(thr_table,lowfreq=10)

one_corpus_num<-rowSums(as.matrix(one_table[one_corpus,]))
one_corpus_table<-data.frame(Word=names(one_corpus_num),frequency=one_corpus_num)
one_corpus_sort<-one_corpus_table[order(-one_corpus_table$frequency),]
head(one_corpus_sort)
```

```
##      Word frequency
## just just      2484
## like like      2259
## will will      2162
## one   one      2098
## get   get      1898
## can   can      1886
```

```
two_corpus_num<-rowSums(as.matrix(two_table[two_corpus,]))
two_corpus_table<-data.frame(Word=names(two_corpus_num),frequency=two_corpus_num)
two_corpus_sort<-two_corpus_table[order(-two_corpus_table$frequency),]
head(two_corpus_sort)
```

```
##                  Word frequency
## right now   right now       230
## cant wait   cant wait       193
## last night last night       168
## dont know   dont know       150
## im going     im going       138
## can get       can get       117
```

```
thr_corpus_num<-rowSums(as.matrix(thr_table[thr_corpus,]))
thr_corpus_table<-data.frame(Word=names(thr_corpus_num),frequency=thr_corpus_num)
thr_corpus_sort<-thr_corpus_table[order(-thr_corpus_table$frequency),]
head(thr_corpus_sort)
```

```
##                                Word frequency
## cant wait see         cant wait see        35
## happy mothers day happy mothers day        33
## let us know             let us know        27
## happy new year       happy new year        18
## im pretty sure       im pretty sure        18
## dont even know       dont even know        15
```

Extract the word and frequency of N-grams.

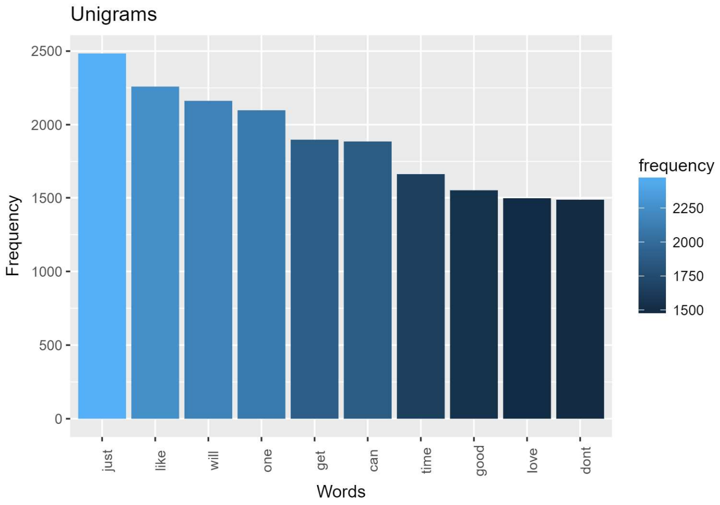
### Plot graph

```
library(ggplot2)
```

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

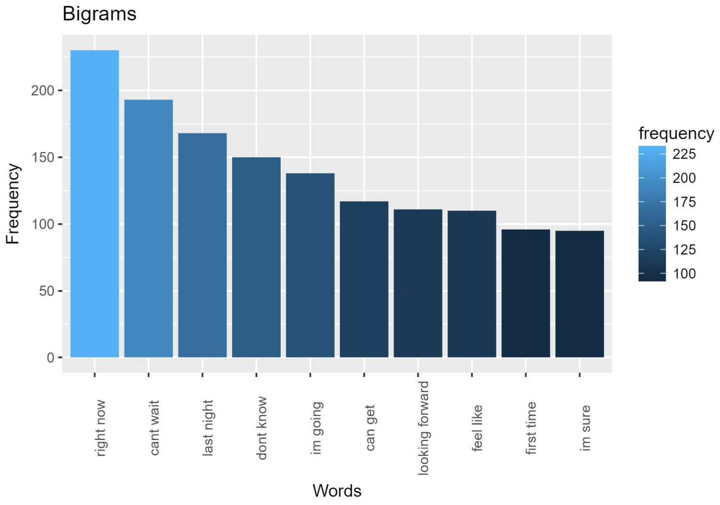
```
## The following object is masked from 'package:NLP':
##
##     annotate
```

```
one_g<-ggplot(one_corpus_sort[1:10,],aes(x=reorder(Word,-frequency),y=frequency,fill=frequency))
one_g<-one_g+geom_bar(stat="identity")
one_g<-one_g+labs(title="Unigrams",x="Words",y="Frequency")
one_g<-one_g+theme(axis.text.x=element_text(angle=90))
one_g
```



```
two_g<-ggplot(two_corpus_sort[1:10,],aes(x=reorder(Word,-frequency),y=frequency,fill=frequency))
two_g<-two_g+geom_bar(stat="identity")
```

```
two_g<-two_g+labs(title="Bigrams",x="Words",y="Frequency")
two_g<-two_g+theme(axis.text.x=element_text(angle=90))
two_g
```



```
thr_g<-ggplot(thr_corpus_sort[1:10,],aes(x=reorder(Word,-frequency),y=frequency,fill=frequency))
thr_g<-thr_g+geom_bar(stat="identity")
thr_g<-thr_g+labs(title="Trigrams",x="Words",y="Frequency")
thr_g<-thr_g+theme(axis.text.x=element_text(angle=90))
thr_g
```