Milestone Report Submission

Shane Kao

Monday, February 16, 2015

Download Data and Import Data

First of all, we demonstrate that how downloaded the data and successfully loaded it in R as character vector.

```
setwd("C:/Users/asus/Downloads")
destination_file <- "Coursera-SwiftKey.zip"
source_file <- "http://d396qusza40orc.cloudfront.net/dsscapstone/dataset/Coursera-SwiftKey.zip"
if(!destination_file%in%list.files(getwd())){
        download.file(source_file, destination_file)
        unzip(destination_file, list = TRUE )
}
list.files("final")
```

```
## [1] "de_DE" "en_US" "fi_FI" "ru_RU"
```

```
setwd("final/en_US")
file.info(list.files())[c("en_US.blogs.txt","en_US.twitter.txt","en_US.news.txt"),]
```

```
##                        size isdir mode               mtime
## en_US.blogs.txt   210160014 FALSE  666 2014-07-22 10:13:06
## en_US.twitter.txt 167105338 FALSE  666 2014-07-22 10:12:58
## en_US.news.txt    205811885 FALSE  666 2015-02-15 23:01:23
##                                 ctime               atime exe
## en_US.blogs.txt   2015-02-15 19:12:37 2015-02-15 19:12:37  no
## en_US.twitter.txt 2015-02-15 19:12:22 2015-02-15 19:12:22  no
## en_US.news.txt    2015-02-15 19:12:29 2015-02-15 19:12:29  no
```

```
twitter=readLines("en_US.twitter.txt",encoding="UTF-8")
summary(twitter)
```

```
##    Length     Class      Mode
##   2360148 character character
```

```
blogs=readLines("en_US.blogs.txt",encoding="UTF-8")
summary(blogs)
```

```
##    Length     Class      Mode
##    899288 character character
```

```
news=readLines("en_US.news.txt",encoding="UTF-8")
summary(news)
```

```
##    Length     Class      Mode
##   1010242 character character
```

Basic Statistics

First of all, we count the words per item (line) and summarise the distibution of these three files.

```r
summary(nchar(twitter,allowNA=TRUE))
```

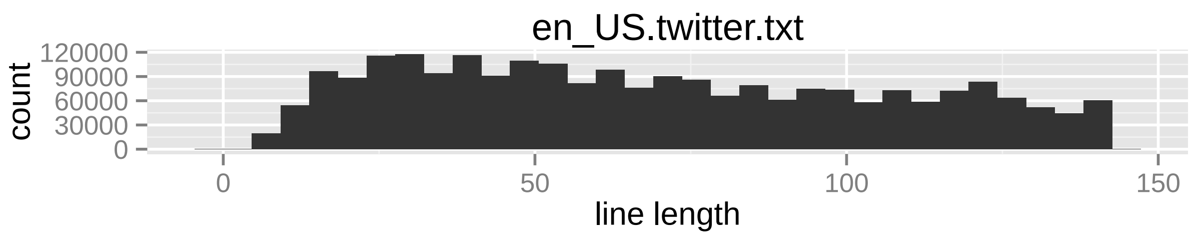
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00   37.00   64.00   68.68  100.00  140.00
```

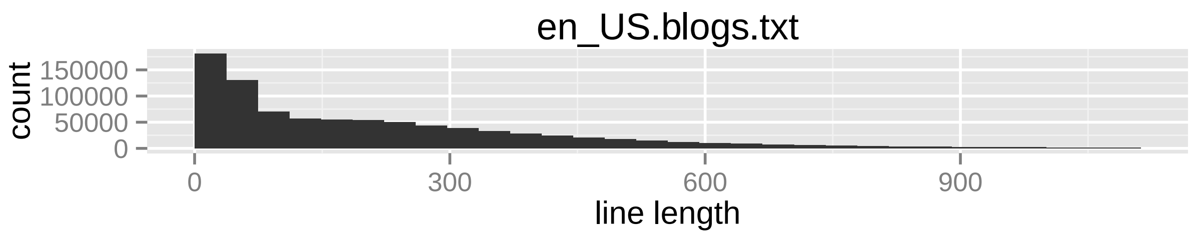
```r
summary(nchar(blogs,allowNA=TRUE))
```

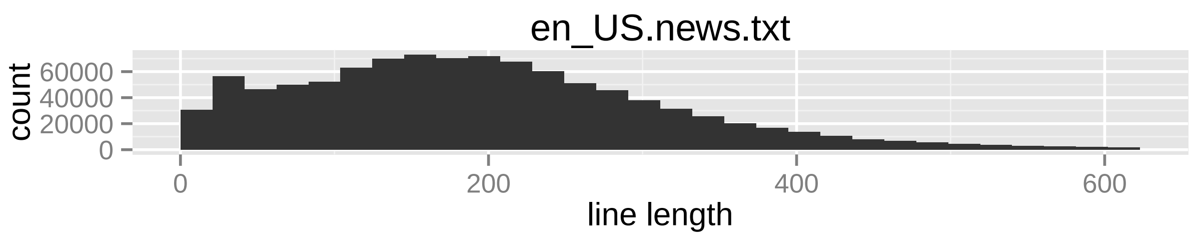
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1      47     156     230     329   40830
```

```r
summary(nchar(news,allowNA=TRUE))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0   110.0   185.0   201.2   268.0 11380.0
```







Data Preprpocessing

- Remove lines containing invalid multibyte

- Remove lines containing numbers
- Translate words to lower case
- Remove profanity
- Remove punctuation

```
clean_data=function(x){
        x<-x[!is.na(nchar(x,allowNA=TRUE))]
        x<-x[grep("[0-9]",x,invert=TRUE)]
        x<-apply(cbind(x),1,tolower)
        x<-x[grep("fuck|shit|ass|suck|dick",x,invert=TRUE)]
        x<-gsub("[^a-z\ ]","",x)
        write.table(x,paste0(x,"_clean.txt"),row.names=FALSE,col.names=FALSE)
}
```

The frequencies of n-grams

Top 10 of 1-grams

```
head(TDM_dense[order(TDM_dense$count,decreasing=TRUE),c("Terms","count")],10)
```

```
##        Terms  count
## 137483   the 213303
## 16822    and 125526
## 137389  that  54866
## 58422    for  46948
## 154675   you  36377
## 151599  with  35250
## 148695   was  26672
## 67221   have  26220
## 18580    are  25181
## 29421    but  24679
```

Top 10 of 2-grams

```
head(TDM_bigram_dense[order(TDM_bigram_dense$count,decreasing=TRUE),c("Terms","count")],10)
```

```
##           Terms count
## 883001   of the 20591
## 626948   in the 17542
## 1315775  to the 10221
## 897639   on the  9110
## 1306575   to be  8714
## 475601  for the  7244
## 82073   and the  6712
## 119823   at the  6064
## 619930     in a  5802
## 652006     is a  5223
```

Top 10 of 3-grams

```r
head(TDM_trigram_dense[order(TDM_trigram_dense$count,decreasing=TRUE),c("Terms","count")],10)
```

```
##                  Terms count
## 1952939    one of the  1764
## 34674        a lot of  1609
## 2826055       to be a   884
## 1288709     i want to   795
## 376524     be able to   794
## 2010263    out of the   769
## 1079359   going to be   759
## 2410777   some of the   742
## 321545     as well as   732
## 2625199 the fact that   673
```

Discussion

I'm suffering by the slowness of `tm` package, so I use subset of each files to investigate the frequencies of n-grams, and we can use these frequency n-grams to test the model, because it seems like people use these words or phrase more often, we want the model has good performance to predict the next word.