Milestone-First

Oscar Villa

November 28, 2016

Introduction:

The goal of this project is just to display that I've gotten used to working with the data and that I'm on track to create my own prediction algorithm.

The data sets:

There are three data sets: the one from twitter, the one from blogs and the one from news. All they are text and each one it's so big that generates a big work load and accopies a lot of RAM. It's because of this that we'll t take jus samples from them.

Importing and sampling data sets:

1. Loading libraries: First of all, load needed libraries and enabling multicore work. Really we'll need it.

```
## Loading packages
require(quanteda)
library(doMC)
library(text2vec)
## Enabling multicore
registerDoMC(cores = 4)
```

2. Loading data: Will load the data sets, but because of their size, keep just a random sample of 0.10 out of the original.

```
twitter <- readLines("/home/oscar/Documents/DSSCapstone/en_US/en_US.twitter.txt")
news <- readLines("/home/oscar/Documents/DSSCapstone/en_US/en_US.news.txt")
blogs <- readLines("/home/oscar/Documents/DSSCapstone/en_US/en_US.blogs.txt")

twitter <- twitter[sample(seq(1, length(twitter)), size = length(twitter) * 0.1)]
news <- news[sample(seq(1, length(news)), size = length(news) * 0.1)]
blogs <- blogs[sample(seq(1, length(blogs)), size = length(blogs) * 0.1)]

head(twitter, 2)
```

```
## [1] "Yes. Ended up doing factory reset and abandoning ADW Launcher EX which seemed to cause lag/failu
## [2] "3:00 in the morning really?"
```

```
head(news, 2)
```

```
## [1] "It's unclear how the much smaller D.C. order might affect hiring. Oregon Iron Works employs 400
## [2] "According to Soriano, investigators learned that the burglaries followed a pattern: The intrude
```

```
head(blogs, 2)
```

```
## [1] "Essentially, the old-school boss of the corporation (Bigweld) has been booted to the side and th
## [2] ""Values are not central to today's world," or so proclaimed the author of an article published
```

3. For the aim of avoid memory issues we'll parallelize the work. So, first, split the samples in chunks:

```
twitter <- split_into(twitter, 100)
news <- split_into(news, 100)
blogs <- split_into(blogs, 100)
```

4. Will end to enable the multicore for run the quanteda's functions on parallel for each (with foreach) of the chunks

4.1. First for onegrams: I know that this work could be done recursively with just one function which takes the ngram and the data set as arguments, but now it's not straightforward for me.

```
library(doMC)
registerDoMC(cores = 4)
library(foreach)
library(doParallel)

twitterdfm <- foreach(i=1:length(twitter), .combine = rbind, .packages = c("quanteda")) %dopar% {
splitsn <- as.vector(twitter[[i]])
dfm <- dfm(as.character(splitsn), verbose = FALSE, what = c("word"), removeNumbers = T,removePunct = T,
}
```

```
##
```

```
newsdfm <- foreach(i=1:length(news), .combine = rbind, .packages = c("quanteda")) %dopar% {
splitsn <- as.vector(news[[i]])
dfm <- dfm(as.character(splitsn), verbose = FALSE, what = c("word"), removeNumbers = T, removePunct = T
}
```

```
##
```

```
blogsdfm <- foreach(i=1:length(blogs), .combine = rbind, .packages = c("quanteda")) %dopar% {
splitsn <- as.vector(blogs[[i]])
dfm <- dfm(as.character(splitsn), verbose = FALSE, what = c("word"), removeNumbers = T, removePunct = T
}
```

```
##
```

```
head(twitterdfm)
```

```
## Document-feature matrix of: 236,014 documents, 92,394 features.
## (showing first 6 documents and first 6 features)
##        features
## docs    yes up doing and which to
##   text1   1  1     1   1     1  1
##   text2   0  0     0   0     0  0
##   text3   0  0     0   0     0  0
##   text4   0  0     0   0     0  1
##   text5   0  0     0   0     0  1
##   text6   0  0     0   0     0  1
```

```
head(newsdfm)
```

```
## Document-feature matrix of: 101,024 documents, 89,668 features.
## (showing first 6 documents and first 6 features)
##        features
## docs    it's how the much order might
##   text1    1   1   1    1     1     1
##   text2    0   0   3    0     0     0
##   text3    0   0   3    0     0     0
```

```
##   text4    0   0   0    0     0     0
##   text5    0   0   8    0     0     0
##   text6    0   0   1    0     0     0
```

```
head(blogsdfm)
```

```
## Document-feature matrix of: 89,928 documents, 93,786 features.
## (showing first 6 documents and first 6 features)
##        features
## docs    the old school of has been
##   text1   5   1      1  1   3    1
##   text2  10   0      0  4   0    0
##   text3   0   0      0  0   0    0
##   text4   1   0      0  0   0    0
##   text5   5   0      0  0   0    0
##   text6   0   0      0  0   0    0
```

```
alldfm <- rbind(twitterdfm, newsdfm, blogsdfm)
```

```
##
```

```
head(alldfm)
```

```
## Document-feature matrix of: 426,966 documents, 181,741 features.
## (showing first 6 documents and first 6 features)
##        features
## docs    yes up doing and which to
##   text1   1  1     1   1     1  1
##   text2   0  0     0   0     0  0
##   text3   0  0     0   0     0  0
##   text4   0  0     0   0     0  1
##   text5   0  0     0   0     0  1
##   text6   0  0     0   0     0  1
```

```
library(doMC)
registerDoMC(cores = 4)
library(foreach)
library(doParallel)

twitterdfmbi <- foreach(i=1:length(twitter), .combine = rbind, .packages = c("quanteda")) %dopar% {
splitsn <- as.vector(twitter[[i]])
ng <- dfm(tokenize(splitsn, ngrams = 2 , verbose = FALSE, removeNumbers = T, removePunct = T, removeSym
}
```

```
##
```

```
newsdfmbi <- foreach(i=1:length(news), .combine = rbind, .packages = c("quanteda")) %dopar% {
splitsn <- as.vector(news[[i]])
ng <- dfm(tokenize(splitsn, ngrams = 2 , verbose = FALSE, removeNumbers = T, removePunct = T, removeSym
}
```

```
##
```

```
blogsdfmbi <- foreach(i=1:length(blogs), .combine = rbind, .packages = c("quanteda")) %dopar% {
splitsn <- as.vector(blogs[[i]])
ng <- dfm(tokenize(splitsn, ngrams = 2 , verbose = FALSE, removeNumbers = T, removePunct = T, removeSym
}
```

```
##
```

```
head(twitterdfmbi)
```

```
## Document-feature matrix of: 236,014 documents, 1,056,629 features.
## (showing first 6 documents and first 6 features)
##        features
## docs    in_the for_the the_most in_a I_want want_to
##   text1      0       0        0    0      0       0
##   text2      1       0        0    0      0       0
##   text3      0       0        0    0      0       0
##   text4      0       1        1    1      0       0
##   text5      0       0        0    1      1       1
##   text6      0       0        0    0      0       0
```

```
head(newsdfmbi)
```

```
## Document-feature matrix of: 101,024 documents, 1,235,891 features.
## (showing first 6 documents and first 6 features)
##        features
## docs    that_the in_the more_than to_the of_the the_season
##   text1        0      0         0      0      0          0
##   text2        1      1         0      0      0          0
##   text3        0      0         1      1      1          1
##   text4        0      0         0      0      0          0
##   text5        0      0         2      1      1          0
##   text6        0      0         0      0      0          0
```

```
head(blogsdfmbi)
```

```
## Document-feature matrix of: 89,928 documents, 1,264,844 features.
## (showing first 6 documents and first 6 features)
##        features
## docs    of_the has_been to_the and_the that_it it_is
##   text1      1        1      1       1       1     1
##   text2      0        0      0       0       0     1
##   text3      0        0      0       0       0     0
##   text4      0        0      0       0       0     0
##   text5      0        0      1       0       0     0
##   text6      0        0      0       0       0     0
```

```
alldfmbi <- rbind(twitterdfmbi, newsdfmbi, blogsdfmbi)
```

```
##
```

```
head(alldfmbi)
```

```
## Document-feature matrix of: 426,966 documents, 2,960,091 features.
## (showing first 6 documents and first 6 features)
##        features
## docs    in_the for_the the_most in_a I_want want_to
##   text1      0       0        0    0      0       0
##   text2      1       0        0    0      0       0
##   text3      0       0        0    0      0       0
##   text4      0       1        1    1      0       0
##   text5      0       0        0    1      1       1
##   text6      0       0        0    0      0       0
```