First steps in Text Mining the Quran

Abdulbaqi Sharaf

Wednesday, April 29, 2015

In the previous tutorial we created the data frame that hold the Arabic Quran called `q`

```
str(q)
```

```
## 'data.frame':    6236 obs. of  3 variables:
##  $ sura: int  1 1 1 1 1 1 1 2 2 2 ...
##  $ aya : int  1 2 3 4 5 6 7 1 2 3 ...
##  $ text: chr  "ما" "الرحمن الرحيم" "الحمد لله رب العالمين" "الرحمن الرحيم" "بسم الله الرحمن الرحيم"
لك يوم الدين" ...
```

Text Mining Package

I wanted to experiment a bit with the `tm` package. Please install and load the package.

```
library(tm)
```

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

The first step is to create a corpus consisting of the raw Arabic verses as `VectorSource`

```
qCorpus = Corpus(VectorSource(q$text))
```

Lets `inspect` the content of this Corpus

```
inspect(qCorpus[1:5])
```

```
## <<VCorpus (documents: 5, metadata (corpus/indexed): 0/0)>>
##
## [[1]]
## <<PlainTextDocument (metadata: 7)>>
## بسم الله الرحمن الرحيم
##
## [[2]]
## <<PlainTextDocument (metadata: 7)>>
## الحمد لله رب العالمين
##
## [[3]]
## <<PlainTextDocument (metadata: 7)>>
## الرحمن الرحيم
##
## [[4]]
## <<PlainTextDocument (metadata: 7)>>
## مالك يوم الدين
##
## [[5]]
## <<PlainTextDocument (metadata: 7)>>
## إياك نعبد وإياك نستعين
```

We will do some more annotation work using `meta` later. For now, let us create term document matrix

```
qTerms = DocumentTermMatrix(qCorpus)
qTerms
```

```
## <<DocumentTermMatrix (documents: 6236, terms: 14766)>>
## Non-/sparse entries: 63181/92017595
## Sparsity           : 100%
## Maximal term length: 11
## Weighting          : term frequency (tf)
```

This produces a long matrix of documents (i.e., verses) against Quranic terms. Let us for example see a portion of this matrix by looking into documents 1 to 7 (i.e., sura Fateha) and terms say 1000 to 1005

```
inspect(qTerms[1:7,1000:1005])
```

```
## <<DocumentTermMatrix (documents: 7, terms: 6)>>
## Non-/sparse entries: 0/42
## Sparsity           : 100%
## Maximal term length: 6
## Weighting          : term frequency (tf)
##
##     Terms
## Docs أعيدوا أعيذها أعين أعينكم أعينهم أعينهن
##    1      0      0    0      0      0      0
##    2      0      0    0      0      0      0
##    3      0      0    0      0      0      0
##    4      0      0    0      0      0      0
##    5      0      0    0      0      0      0
##    6      0      0    0      0      0      0
##    7      0      0    0      0      0      0
```

This tells us that none of these five terms appears in any of the first 7 documents. Sparsity is a known issue in document term matrices.

Some operations of Document Term Matrics

Lets us find some common terms in the Quran. What are terms used 100 or more times in the Quran?

```
findFreqTerms(qTerms,100)
```

```
##  [1] "الذي"   "الدنيا"    "الحق"    "الأرض"     "إلا"     "إذا"
##  [7] "النار"    "الله"   "الكتاب" "السماوات"   "السماء"    "الذين"
## [13] "إنه"     "إنما"     "إنا"    "آمنوا"     "إلى"    "الناس"
## [19] "بما"      "بعد"    "بالله"    "أيها"    "أولئك"     "إني"
## [25] "ربنا"     "ربكم"     "ربك"     "ذلك"     "خير"     "حتى"
## [31] "عليم"    "عليكم"     "على"     "عذاب"     "شيء"     "ربهم"
## [37] "فلما"      "فلا"     "فإن"     "عند"    "عليهم"    "عليه"
## [43] "قوم"      "قبل"    "قالوا"     "قال"     "فيها"     "فيه"
## [49] "لله"      "لكم"    "كنتم"    "كفروا"    "كانوا"     "كان"
## [55] "هذا"     "موسى"     "منهم"    "منكم"      "مما"     "لهم"
## [61] "ولا"     "وإن"    "والله"   "والذين"   "والأرض"    "وإذا"
## [67] "وهم"      "ومن"     "وما"     "ولو"     "ولكن"    "ولقد"
## [73] "يوم"     "يشاء"     "وهو"
```

Interesting to see prophet Musa (Moses) “موسى” among the list.
Note that since we did not do any stemming root words are repeated with various affixes as different words.

Even we can create a list of most freq terms and store it in a data frame

```
freq = sort(colSums(as.matrix(qTerms)),decreasing = T)
head(freq, 10)
```

```
##  ومن   لهم   إلى   قال   وما   ولا   إلا   على الذين   الله
##  2153   810   670   664   658   646   416   405   373   342
```

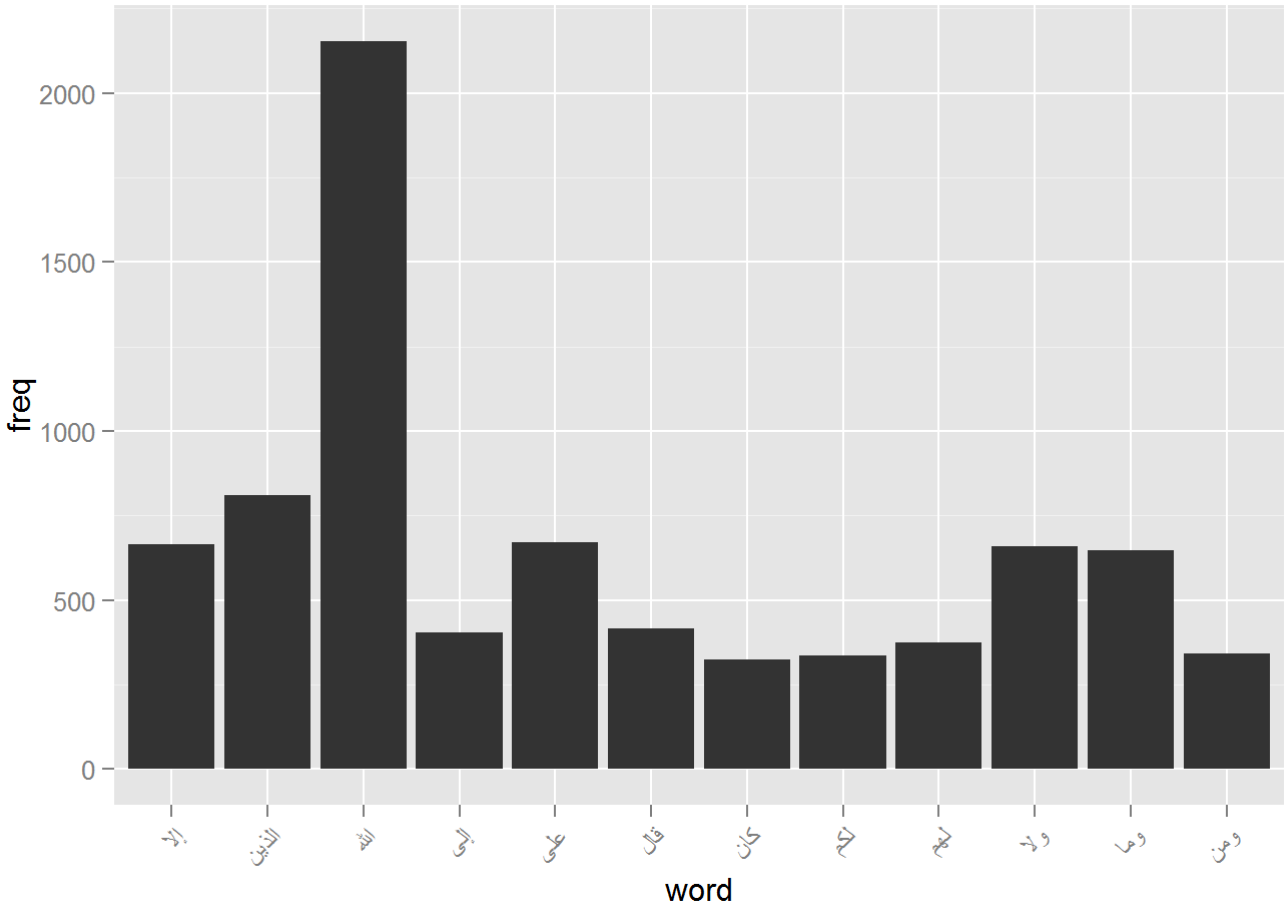
```
wf = data.frame(word=names(freq), freq=freq)
```

Why not plot them using `ggplot2` package?

```
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
##
## The following object is masked from 'package:NLP':
##
##     annotate
```

```
#take the most freq in a separate data frame
wfplot = subset(wf,freq>300)
ggplot(wfplot, aes(word, freq)) +
  geom_bar(stat="identity")+
  theme(axis.text.x=element_text(angle=45, hjust = 1))
```



No Wonder, Allah الله is the most frequent word. May HE be exalted!

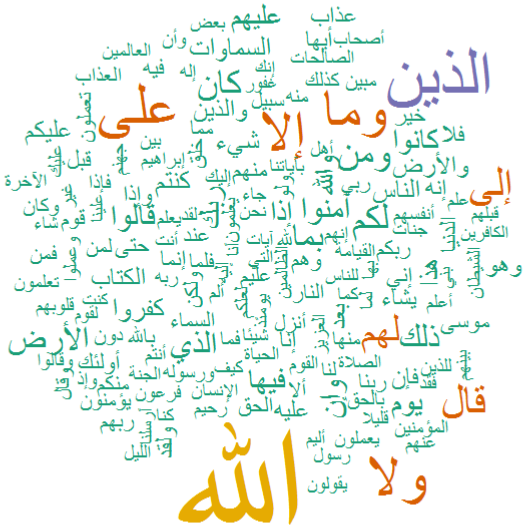
Word Cloud

Now let us do some more cool visualization with Word Cloud using the package `wordcloud` .Please review the package and adjust various parameters to choose the right scale and color brewer and percentage of words to rotate.

```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
#I will set a seed so you can reproduce this result
set.seed(114)
wordcloud(names(freq), freq, min.freq=50, scale=c(5,.5),colors=brewer.pal(6,"Dark2"),
rot.per=0.2)
```



Alayws the beatiful name ALLAH (الله) pops up in your face! Exalted be He.

Word Length

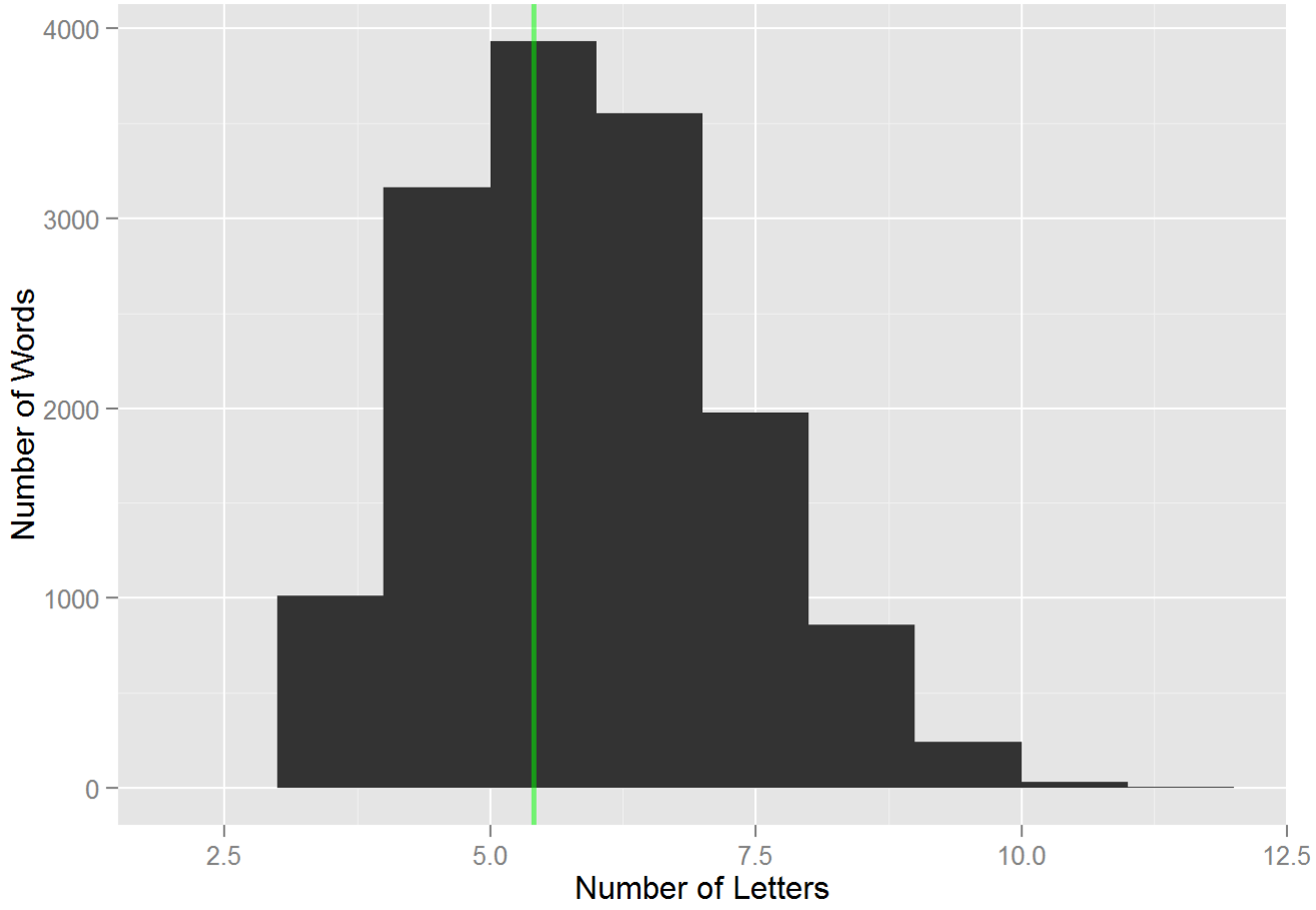
We would like to know more about the word length in the Quran.

First lets get all words in a data frame `words` and word length in `wLen`

```
words = as.matrix(colnames(qTerms))
wLen = data.frame(nletters=nchar(words))
```

Let us produce some visualization out of this.

```
ggplot(wLen, aes(x=nletters))+
  geom_histogram(binwidth=1) +
  geom_vline(xintercept=mean(nchar(words)),
             colour="green", size=1, alpha=.5)+
  labs(x="Number of Letters", y="Number of Words")
```



This shows that on average word sizes are close to 5 letters. Remember we are not talking here about root words, rather raw words with all prefixes and suffixes.

Letter frequencies

Since we gone that far, let us conclude with analyzing the frequency of letters. First a number of packages need to be installed.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringr)
library(qdap)
```

```
## Loading required package: qdapDictionaries
## Loading required package: qdapRegex
##
## Attaching package: 'qdapRegex'
##
## The following objects are masked from 'package:dplyr':
##
##     escape, explain
##
## The following object is masked from 'package:ggplot2':
##
##     %+%
##
## Loading required package: qdapTools
##
## Attaching package: 'qdapTools'
##
## The following object is masked from 'package:dplyr':
##
##     id
##
## WARNING: Rtools is required to build R packages, but is not currently installed.
##
## Please download and install Rtools 3.1 from http://cran.r-project.org/bin/windows/
Rtools/ and then run find_rtools().
##
## Attaching package: 'qdap'
##
## The following object is masked from 'package:dplyr':
##
##     %>%
##
## The following objects are masked from 'package:tm':
##
##     as.DocumentTermMatrix, as.TermDocumentMatrix
##
## The following object is masked from 'package:base':
##
##     Filter
```

```
letter = str_split(words,"")
letter=sapply(letter, function(x) x[-1])
letter = unlist(letter)
letter = dist_tab(letter)
```

So, `letter` is a nice data frame that gives a list of letters with their frequency and cumulitive freq percentages. Let us produce a graph out of it

```
letterMutate = mutate(letter,Letter=factor(letter$interval, levels=letter$interval[or
der(letter$freq)]))
ggplot(letterMutate, aes(letterMutate$Letter, weight=percent)) +
  geom_bar()+
  coord_flip()+
  ylab("Proportion")+
  xlab("Letter")+
  scale_y_continuous(breaks=seq(0,12,2),
                     label=function(x) paste0(x,"%"),
                     expand=c(0,0), limits=c(0,12))
```