Exercise 1

Ken Benoit

26 September 2014

Using quanteda in R

This take-home exercise is designed to get you working with quanteda (http://github.com/kbenoit/quanteda). The focus will be on exploring the package and getting some texts into the **corpus** object format. quanteda (http://github.com/kbenoit/quanteda) package has several functions for creating a corpus of texts which we will use in this exercise.

1. Getting Started.

   You can use R or Rstudio for these exercises. You will first need to install the package, using:

   ```
   # needs the devtools package for this to work
   if (!require(devtools)) install.packages("devtools", dependencies=TRUE)
   # be sure to install the latest version from GitHub, using dev branch:
   devtools::install_github("quanteda", username="kbenoit", dependencies=TRUE, ref="dev")
   # and quantedaData
   devtools::install_github("quantedaData", username="kbenoit")
   ```

2. Exploring **quanteda** functions.

   You can try running `demo(quanteda)`, and also use the `example()` function for any function in the package, to run the examples and see how the function works. Of course you should also browse the documentation, especially `?corpus` to see the structure and operations of how to construct a corpus.

3. Making a corpus and corpus structure

   1. From a vector of texts already in memory.

      The simplest way to create a corpus is to use a vector of texts already present in R's global environment. Some text and corpus objects are built into the package, for example `inaugTexts` is the UTF-8 encoded set of 57 presidential inaugural addresses. Try using `corpus()` on this set of texts to create a corpus.

      Once you have constructed this corpus, use the `summary()` method to see a brief description of the corpus. The names of the character vector `inaugTexts` should have become the document names.

   2. From a directory of text files.

      The `corpus()` function can take as its main argument the name of a directory, if you wrap the path to the directory within a `directory()` call. (See `?directory` for an example.) If you call `directory()` with no arguments, then it should allow you to choose the directory interactively (you will need to have installed the `tcltk2` package first though.)

Here you are encouraged to select any directory of plain text files of your own.
How did it work? Try using `docvars()` to assign a set of document-level variables.

Note that if you document level metadata in your filenames, then this can be automatically parsed by `corpus.directory()` into `docvars`.

```
require(quanteda)
```

```
## Loading required package: quanteda
```

```
mydir <- directory("~/Dropbox/QUANTESS/corpora/ukManRenamed")
mycorpus <- corpus(mydir)
summary(mycorpus, 5)
```

```
## Corpus consisting of 101 documents.
##
##                  Text Types Tokens Sentences docvar1 docvar2 docvar3
##  UK_natl_1945_en_Con  1578   6095       275      UK    natl    1945
##  UK_natl_1945_en_Lab  1258   4975       241      UK    natl    1945
##  UK_natl_1945_en_Lib  1060   3377       158      UK    natl    1945
##  UK_natl_1950_en_Con  1800   7413       381      UK    natl    1950
##  UK_natl_1950_en_Lab  1342   4879       275      UK    natl    1950
##  docvar4 docvar5
##       en     Con
##       en     Lab
##       en     Lib
##       en     Con
##       en     Lab
##
## Source:  /Users/kbenoit/Dropbox/Classes/QTA NYU/Exercises/Exercise 1/* on x86_64 by kbenoi
t.
## Created: Fri Sep 26 12:39:15 2014.
## Notes:   .
```

3. From a zipped file. For this, you can try downloading the zipped file of Irish budget speeches available from http://www.kenbenoit.net/courses/nyu2014qta/iebudget2010.zip (http://www.kenbenoit.net/courses/nyu2014qta/iebudget2010.zip). Then execute this code:

```
myzipfiles <- zipfiles()   # allows you to locate the file interactively
mycorpus <- corpus(myzipfiles,
                   docvarnames=c("year", "debate", "seq", "fname", "lname", "party"))
summary(mycorpus, 5)
```

4. If you are familiar with the **tm** text package for R, it is also now possible to import one of its `VCorpus` objects directly, using the `corpus.VCorpus()` method.
This is a feature I just added, so I am eager for you to test it.

4. Explore some phrases in the text.

   You can do this using the `kwic` (for “key-words-in-context”) to explore a specific word or phrase.

```
kwic(inaugCorpus, "terror", 3)
```

```
##                                                     preword      word
##    [1797-Adams, 1183]                    or violence, by   terror,
## [1933-Roosevelt, 100] nameless, unreasoning, unjustified    terror
## [1941-Roosevelt, 252]                    by a fatalistic   terror,
##   [1961-Kennedy, 763]               uncertain balance of   terror
##   [1961-Kennedy, 872]                     instead of its  terrors.
##    [1981-Reagan, 691]                 Americans from the  terror
##   [1981-Reagan, 1891]                 those who practice terrorism
##   [1997-Clinton, 929]                  the fanaticism of   terror.
##  [1997-Clinton, 1462]             strong defense against   terror
##    [2009-Obama, 1433]                   aims by inducing    terror
##                                               postword
##    [1797-Adams, 1183] intrigue, or venality,
## [1933-Roosevelt, 100] which paralyzes needed
## [1941-Roosevelt, 252] we proved that
##   [1961-Kennedy, 763] that stays the
##   [1961-Kennedy, 872] Together let us
##    [1981-Reagan, 691] of runaway living
##   [1981-Reagan, 1891] and prey upon
##   [1997-Clinton, 929] And they torment
##  [1997-Clinton, 1462] and destruction. Our
##    [2009-Obama, 1433] and slaughtering innocents,
```

   Try substituting your own search terms, or working with your own corpus.

5. Create a document-feature matrix, using `dfm`. First, read the documentation using `?dfm` to see the available options.

```
mydfm <- dfm(inaugCorpus, stopwords=TRUE)
```

```
## Creating dfm from a corpus: ...  removing stopwords ...  done.
```

```
dim(mydfm)
```

```
## [1]   57 9087
```

```
topfeatures(mydfm, 20)
```

```
##       will     people government         us        can       upon
##        871        564        561        476        470        371
##       must        may      great     states      shall      world
##        363        338        334        331        314        305
##    country      every     nation      peace        one        new
##        294        291        287        253        244        241
##      power     public
##        232        223
```

Experiment with different `dfm` options, such as `stem=TRUE`. The function `trimdfm()` allows you to reduce the size of the dfm following its construction.

Grouping on a variable is an excellent feature of `dfm()`, in fact one of my favorites.
For instance, if you want to aggregate all speeches by presidential name, you can execute

```
mydfm <- dfm(inaugCorpus, groups="President")
```

```
## Creating dfm from a corpus: ... aggregating by group: President... complete ... done.
```

```
dim(mydfm)
```

```
## [1]   34 9210
```

```
docnames(mydfm)
```

```
##  [1] "Adams"      "Buchanan"   "Bush"       "Carter"     "Cleveland"
##  [6] "Clinton"    "Coolidge"   "Eisenhower" "Garfield"   "Grant"
## [11] "Harding"    "Harrison"   "Hayes"      "Hoover"     "Jackson"
## [16] "Jefferson"  "Johnson"    "Kennedy"    "Lincoln"    "Madison"
## [21] "McKinley"   "Monroe"     "Nixon"      "Obama"      "Pierce"
## [26] "Polk"       "Reagan"     "Roosevelt"  "Taft"       "Taylor"
## [31] "Truman"     "VanBuren"   "Washington" "Wilson"
```

Note that this groups Theodore and Franklin D. Roosevelt together – to separate them we would have needed to add a firstname variable using `docvars()` and grouped on that as well.

6. Explore the ability to subset a corpus.

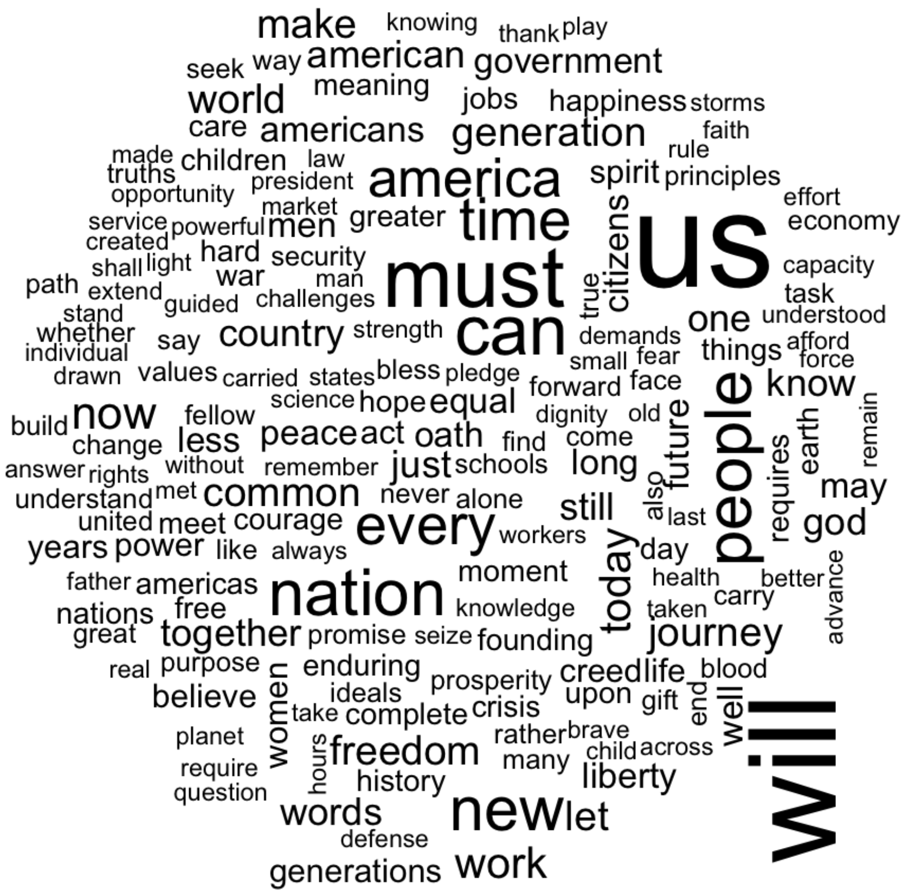
   There is a `subset()` method defined for a corpus, which works just like R's normal `subset()` command. This provides an easy method to send specific documents to downstream functions, like `dfm()`, which will be useful workaround until I implement a full set of subsetting and indexing features for the `dfm` class object.

   For instance if you want a wordcloud of just Obama's two inagural addresses, you would need to subset the corpus first:

```
obamadfm <- dfm(subset(inaugCorpus, President=="Obama"), stopwords=TRUE)
```

```
## Creating dfm from a corpus: ...  removing stopwords ...  done.
```

```
plot(obamadfm)
```



Bug reports

quanteda is a work in progress. Please send me suggestions, bug reports, etc. so that I can improve it. You can email these directly to kbenoit@lse.ac.uk (mailto:kbenoit@lse.ac.uk).

In addition, if you are having trouble with importing your texts into a corpus, I welcome you to send me a set of your texts and I will write the code for you (and possibly add to the functionality of **quanteda** to make this possible). I am particularly interested in: * alternative formats - csv, pdf, Word, XML, etc. * non-English languages - encoding is one of the issues we plan to tackle * interesting/non-standard document units * large volumes of text, to see **quanteda**'s functionality scales.