newscrapeR
A data mining tool for R

Philipp Burckhardt

December 26, 2012

1 Introduction

The newscrapeR package provides a YQL-based tool for data mining. The idea of YQL (an acronym for Yahoo! Query Language) is to understand the Internet as a big database. Therefore, YQL is designed in the style of SQL, the standard language for relational databases. One of the greatest advantages of YQL is that Yahoo! works as a proxy for your queries allowing cross-domain communication.

With YQL as an underlying framework, this package aims to provide the user with an easy-to-use interface to build up huge collections of articles, in a format ready to be analyzed via common statistical methods. Therefore, export methods for formats such as data.frames or text mining Corpora are provided.

In detail, the newscrapeR package grants scanning of various news sources (such as newspapers) and the extraction of the current articles from their websites. The article contents are stored in plain-text format, which means: all contaminating information such as HTML tags, advertisements, banners etc. are removed. Only the meat remains.

2 Getting Started

To install the package, run the following commands:

```
download.file(url = "http://dl.dropbox.com/u/8439596/newscrapeR_0.7.tar.gz",
    destfile = "newscrapeR.tar.gz")
install.packages("newscrapeR.tar.gz", type = "source", repos = NULL)
```

As usual, the package is loaded by running the following command:

```
library(newscrapeR)

## Loading required package:  rjson
```

3 Workflow

In the following section, the use of the main functions is described. The impatient reader may skip reading and just execute the given commands.

3.1 Creating a newscrapeR object

Creating a newscrapeR object is fairly easy.

```
MyHoustonChronicle <- newscrapeR("Houston Chronicle")
```

With this command, the newly created object MyHoustonChronicle scans the webpages of the Houston Chronicle and stores the articles found on the main page. To inspect how many articles have been fetched, just type the name of your object:

```
MyHoustonChronicle

## Included Sources:
## Houston Chronicle : 15 articles
```

Such an object may consist of just one or a multitude of different sources. So a call like the following is equally valid:

```
MyNewsStand <- newscrapeR(c("Houston Chronicle", "Denver Post", "Chicago Tribune"))
```

This functionality enables the user to build up a highly diversified and flexible article database. To find out which sources are currently supported, just call the following function on your newscrapeR object:

```
available.sources(MyHoustonChronicle)

## Available newscrapeR Sources:
## ----------------------------
## Spiegel Online, URL: http://www.spiegel.de
## Die Welt, URL: http://www.welt.de
## The Independent, URL: http://www.independent.co.uk/
## The Telegraph, URL: http://www.telegraph.co.uk/
## National Post, URL: http://www.nationalpost.com
## Houston Chronicle, URL: http://www.chron.com/
## The Guardian, URL: http://www.guardian.co.uk/
## Chicago Tribune, URL: http://www.chicagotribune.com/
## Baltimore Sun, URL: http://www.baltimoresun.com/
## The Detroit News, URL: http://www.detroitnews.com
## New York Daily News, URL: http://www.nydailynews.com/
## Denver Post, URL: http://www.denverpost.com
```

As you see, there are already a few more newspapers supported besides the Houston Chronicle. This list will be extended in the future, and it will also include articles from the blogosphere and the social media.

Once a newscrapeR object has been created, it is not petrified but can be easily extended to include further news sources.

```
add.sources(MyHoustonChronicle, "Denver Post")
```

3.2 Downloading Articles

The real power of the newscrapeR object lies in its simplicity and its repeated use. With it, it is possible to download the current articles every day without having to face annoying problems like duplicated articles or complicated data management. The download process is as easy as the construction of a new object. Just call:

```
download(MyHoustonChronicle)
```

This call starts the download, thereby updating the object with the newly found articles since all downloaded articles are part of the newscrapeR object.

3.3 Selecting Articles

You may access the downloaded articles by calling:

```
articles(MyHoustonChronicle)
```

There are various options for changing the scope of your search. If you have an object containing multiple sources like our *newsstand*, you might choose only articles from one or many specified sources:

```
articles(MyNewsStand, sources = "Chicago Tribune")
```

You can also scan the articles for a specific keyword:

```
articles(MyNewsStand, keyword = "Obama")
```

This last command scans all the sources of our *newsstand* for articles which contain the given keyword. As expected, you can combine the arguments to scan a specific source for a specific keyword.

Last but not least, you can specify a date range:

```
articles(MyNewsStand, from = "2012-12-20", to = "2012-12-28")
```

3.4 Making Use of newscrapeR Articles

Since newscrapeR is stricly object-oriented (it uses R reference classes), not only the newscrapeR is an object, but also the embedded child objects, like the Article object. An Article contains not only the plain-text, but also additional attributes such as publication date, source and author.

If you want make use of the stored data with methods from other R packages, currently two export functions are provided, which take a list of articles as an argument.

```
my_articles <- articles(MyNewsStand)
ArtListToDF(my_articles)
ArtListToCorpus(my_articles)
```

ArtListToDF exports the article list as a data.frame, whereas ArtListToCorpus exports the list as a text corpus ready to be analyzed with methods from the text mining package "tm".

3.5 Working with multiple objects

Since newscrapeR is stricly object-oriented (it uses R reference classes), you can build as many different newscrapeR objects as you may think of.