Short Introduction to **boilerpipeR**

Mario Annau
mario.annau@gmail.com

November 22, 2024

Abstract

This vignette gives a short introduction to **boilerpipeR**, a package which interfaces the boilerpipe Java library by Kohlschütter et al. (2010). It implements robust heuristics to extract the main content from HTML files, removing unwanted elements like ads, banners, headers and footers.

1 Getting Started

boilerpipeR provides an R interface to the boilerpipe Java library. It implements various robust heuristics to extract the main content from arbitrary web sites. The more sophisticated algorithms included are based on decision trees and have been trained on a real–world data set (retrieved through Google News, `http://news.google.com`).
For a quick content extraction exercise, we first need to retrieve a webpage. After loading the packages for page extraction and retrieval

```
> library(boilerpipeR)
> library(RCurl)
```

we can retrieve the content of from a webpage using **RCurl**:

```
> url <- "https://quantivity.wordpress.com/2012/11/09/multi-asset-market-regimes/"
> content <- getURL(url)
```

The code above retrieves the posting from a quite popular finance blog hosted on Wordpress (`www.wordpress.com`), currently one of the most popular blogging engines on the Internet. An inspection of the retrieved content string reveals a lot of typical HTML markup, including regions like sidebars, headers, etc. (see also Figure 1).

```
> cat(substr(content, 1, 80))
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.or
```

A simple extraction of the HTML–body element and dropping all markup would still include a lot of unnecessary content which can be disturbing for text mining algorithms. We can therefore use one of our default extractors from **boilerpipeR**:

```
> extract <- DefaultExtractor(content)
> cat(substr(extract, 1, 120))
```

```
Multi-Asset MarketÂăRegimes
November 9, 2012
An astute reader suggested reproducing the results from a recent article on
```

2 Implemented Extractors

The list below describes all currently implemented extractors in **boilerpipeR**:

ArticleExtractor A full-text extractor which is tuned towards news articles.

ArticleSentencesExtractor A full-text extractor which is tuned towards extracting sentences from news articles.

CanolaExtractor A full-text extractor trained on a krdwrd.

DefaultExtractor A quite generic full-text extractor.

KeepEverythingExtractor Marks everything as content.

LargestContentExtractor A full-text extractor which extracts the largest text component of a page.

NumWordsRulesExtractor A quite generic full-text extractor solely based upon the number of words per block.

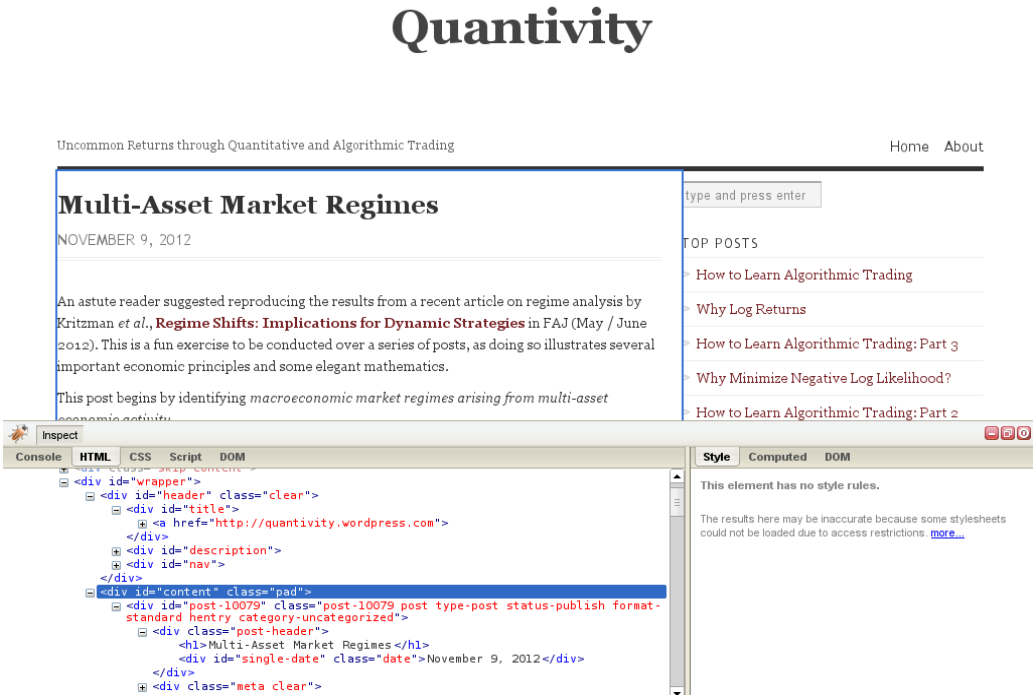

Figure 1: Inspection of a typical Wordpress blog page (https://quantivity.wordpress.com). On the bottom we can see the HTML DOM tree parsed by Firebug (http://getfirebug.com/). Only the main content (blue rectangle) is relevant for text mining purposes and should be extracted.

The following commands show, how the above mentioned extractors can be used:

```
> articleextract <- ArticleExtractor(content)
> articlesentencesextract <- ArticleSentencesExtractor(content)
> canolaextract <- CanolaExtractor(content)
> defaultextract <- DefaultExtractor(content)
> keepeverythingextract <- KeepEverythingExtractor(content)
> largestcontentextract <- LargestContentExtractor(content)
> numwordsrulesextract <- NumWordsRulesExtractor(content)
```

3 Conclusion

This vignette has given a quick introduction to **boilerpipeR**, a package to extract the main content from HTML pages. Although `DefaultExtractor()` fits quite well for most purposes and web pages, each page template may require specialized extraction algorithms or some time to fine tune existing ones. Provided the presented package, the user now has a nice playground to experiment with extraction algorithms from within R. Although **boilerpipeR** is interfacing Java code, it has proven to be very fast and memory efficient—even for larger extraction tasks. Please refer to Kohlschütter et al. (2010) for a detailed explanation of the implented extractors and Kovacic (2011) for a performance comparison of similar text extraction algorithms.

References

Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 441–450, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-889-6. doi: 10.1145/1718487.1718542. URL `https://github.com/kohlschutter/boilerpipe`.

Tomaz Kovacic. Evaluating text extraction algorithms, 06 2011. URL `http://tomazkovacic.com/blog/122/evaluating-text-extraction-algorithms/`.