An Introduction to Text Mining in R

Ingo Feinerer

Introduction

Text mining has gained big interest both in academic research as in business intelligence applications within the last decade. There is an enormous amount of textual data available in machine readable format which can be easily accessed via the Internet or databases. This ranges from scientific articles, abstracts and books to memos, letters, online forums, mailing lists, blogs, and other communication media delivering sensible information.

Text mining is a highly interdisciplinary research field utilizing techniques from computer science, linguistics, and statistics. For the latter R is one of the leading computing environments offering a broad range of statistical methods. However, until recently, R has lacked an explicit framework for text mining purposes. This has changed with the **tm** (Feinerer, 2008; Feinerer et al., 2008) package which provides a text mining infrastructure for R. This allows R users to work efficiently with texts and corresponding meta data and transform the texts into structured representations where existing R methods can be applied, e.g., for clustering or classification.

In this article we will give a short overview of the **tm** package. Then we will present two exemplary text mining applications, one in the field of stylometry and authorship attribution, the second is an analysis of a mailing list. Our aim is to show that **tm** provides the necessary abstraction over the actual text management so that the reader can use his own texts for a wide variety of text mining techniques.

The tm package

The **tm** package offers functionality for managing text documents, abstracts the process of document manipulation and eases the usage of heterogeneous text formats in R. The package has integrated database backend support to minimize memory demands. An advanced meta data management is implemented for collections of text documents to alleviate the usage of large and with meta data enriched document sets. With the package ships native support for handling the Reuters 21578 data set, Gmane RSS feeds, e-mails, and several classic file formats (e.g. plain text, CSV text, or PDFs). The data structures and algorithms can be extended to fit custom demands, since the package is designed in a modular way to enable easy integration of new file formats, readers, transformations and filter operations. **tm** provides easy access to preprocessing and manipulation mechanisms such as whitespace removal, stemming, or conversion between file formats. Further a generic filter architecture is available in order to filter documents for certain criteria, or perform full text search. The package supports the export from document collections to term-document matrices, and string kernels can be easily constructed from text documents.

Wizard of Oz stylometry

The Wizard of Oz book series has been among the most popular children's novels in the last century. The first book was created and written by Lyman Frank Baum, published in 1900. A series of Oz books followed until Baum died in 1919. After his death Ruth Plumly Thompson continued the story of Oz books, but there was a longstanding doubt which was the first Oz book written by Thompson. Especially the authorship of the 15th book of Oz—*The Royal Book of Oz*—has been longly disputed amongst literature experts. Today it is commonly attributed to Thompson as her first Oz book, supported by several statistical stylometric analyses within the last decade.

Based on some techniques shown in the work of Binongo (2003) we will investigate a subset of the Oz book series for authorship attribution. Our data set consists of five books of Oz, three attributed to Lyman Frank Baum, and two attributed to Ruth Plumly Thompson:

The Wonderful Wizard of Oz is the first Oz book written by Baum and was published in 1900.

The Marvelous Land of Oz is the second Oz book by Baum published in 1904.

Ozma of Oz was published in 1907. It is authored by Baum and forms the third book in the Oz series.

The Royal Book of Oz is nowadays attributed to Thompson, but its authorship has been disputed for decades. It was published in 1921 and is considered as the 15th book in the series of Oz novels.

Ozoplaning with the Wizard of Oz was written by Thompson, is the 33rd book of Oz, and was published in 1939.

Most books of Oz, including the five books we use as corpus for our analysis, can be freely downloaded at the Gutenberg Project website (`http://www.gutenberg.org/`) or at the Wonderful Wizard of Oz website (`http://thewizardofoz.info/`).

The main data structure in **tm** is a *corpus* consisting of *text document*s and additional meta data. So-called *reader* functions can be used to read in texts

 ISSN 1609-3631

from various *source*s, like text files on a hard disk. E.g., following code creates the corpus `oz` by reading in the text files from the directory 'OzBooks/', utilizing a standard reader for plain texts.

```
oz <- Corpus(DirSource("OzBooks/"))
```

The directory 'OzBooks/' contains five text files in plain text format holding the five above described Oz books. So the resulting corpus has five elements representing them:

```
> oz
A text document collection with 5 text documents
```

Since our input are plain texts without meta data annotations we might want to add the meta information manually. Technically we use the `meta()` function to operate on the meta data structures of the individual text documents. In our case the meta data is stored locally in explicit slots of the text documents (`type = "local"`):

```
meta(oz, tag = "Author", type = "local") <-
   c(rep("Lyman Frank Baum", 3),
     rep("Ruth Plumly Thompson", 2))
meta(oz, "Heading", "local") <-
   c("The Wonderful Wizard of Oz",
     "The Marvelous Land of Oz",
     "Ozma of Oz",
     "The Royal Book of Oz",
     "Ozoplaning with the Wizard of Oz")
```

E.g. for our first book in our corpus this yields

```
> meta(oz[[1]])
Available meta data pairs are:
  Author       : Lyman Frank Baum
  Cached       : TRUE
  DateTimeStamp: 2008-04-21 16:23:52
  ID           : 1
  Heading      : The Wonderful Wizard of Oz
  Language     : en_US
  URI          : file
    OzBooks//01_TheWonderfulWizardOfOz.txt UTF-8
```

The first step in our analysis will be the extraction of the most frequent terms, similarly to a procedure by Binongo (2003). At first we create a so-called *term-document matrix*, that is a representation of the corpus in form of a matrix with documents as rows and terms as columns. The matrix elements are frequencies counting how often a term occurs in a document.

```
ozMatBaum <- TermDocMatrix(oz[1:3])
ozMatRoyal <- TermDocMatrix(oz[4])
ozMatThompson <- TermDocMatrix(oz[5])
```

After creating three term-document matrices for the three cases we want to distinguish—first, books by Baum, second, the Royal Book of Oz, and third, a book by Thompson—we can use the function `findFreqTerms(mat, low, high)` to compute the terms in the matrix `mat` occurring at least `low` times and at most `high` times (defaults to `Inf`).

```
baum <- findFreqTerms(ozMatBaum, 70)
royal <- findFreqTerms(ozMatRoyal, 50)
thomp <- findFreqTerms(ozMatThompson, 40)
```

The lower bounds have been chosen in a way that we obtain about 100 terms from each term-document matrix. A simple test of intersection within the 100 most common words shows that the Royal Book of Oz has more matches with the Thompson book than with the Baum books, being a first indication for Thompson's authorship.

```
> length(intersect(thomp, royal))
[1] 73
> length(intersect(baum, royal))
[1] 65
```

The next step applies a Principal Component Analysis (PCA), where we use the **kernlab** (Karatzoglou et al., 2004) package for computation and visualization. Instead of using the whole books we create equally sized chunks to consider stylometric fluctuations within single books. In detail, the function `makeChunks(corpus, chunksize)` takes a corpus and the chunk size as input and returns a new corpus containing the chunks. Then we can compute a term-document matrix for a corpus holding chunks of 500 lines length. Note that we use a *binary weighting* of the matrix elements, i.e., multiple term occurrences are only counted once.

```
ozMat <- TermDocMatrix(makeChunks(oz, 500),
           list(weighting = weightBin))
```

This matrix is the input for the Kernel PCA, where we use a standard Gaussian kernel, and request two feature dimensions for better visualization.

```
k <- kpca(as.matrix(ozMat), features = 2)
plot(rotated(k),
     col = c(rep("black", 10), rep("red", 14),
             rep("blue", 10),
             rep("yellow", 6), rep("green", 4)),
     pty = "s",
     xlab = "1st Principal Component",
     ylab = "2nd Principal Component")
```

Figure 1 shows the results. Circles in black are chunks from the first book of Oz by Baum, red circles denote chunks from the second book by Baum, and blue circles correspond to chunks of the third book by Baum. Yellow circles depict chunks from the long disputed 15th book (by Thompson), whereas green circles correspond to the 33rd book of Oz by Thompson. The results show that there is a visual correspondence between the 15th and the 33rd book (yellow and green), this means that both books tend to be authored by Thompson. Anyway, the results also unveil that

there are parts matching with books from Baum.

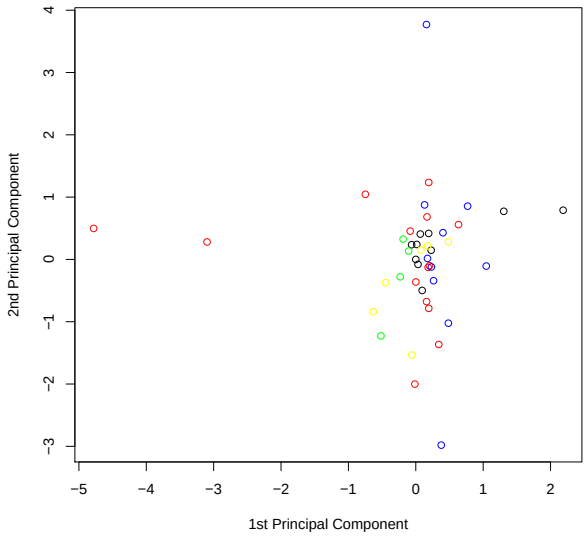

Figure 1: Principal component plot for five Oz books using 500 line chunks.

Similarly, we can redo this plot using chunks of 100 lines. This time we use a weighting scheme called *term frequency inverse document frequency*, a weighting very common in text mining and information retrieval.

```
ozMat <- TermDocMatrix(makeChunks(oz, 100),
          list(weighting = weightTfIdf))
```

Figure 2 shows the corresponding plot using the same colors as before. Again we see that there is a correspondence between the 15th book and the 33rd book by Thompson (yellow and green), but also matching parts with books by Baum.

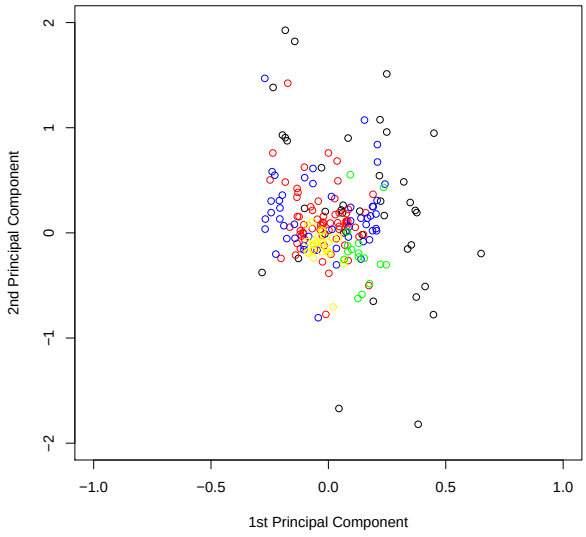

Figure 2: Principal component plot for five Oz books using 100 line chunks.

R-help mailing list analysis

Our second example deals with aspects of analyzing a mailing list. As data set we use the R-help mailing list since its archives are publicly available and can be accessed via an RSS feed or can be downloaded in mailbox format. For the RSS feed we can use the Gmane (Ingebrigtsen, 2008) service, so e.g., with

```
rss <- Corpus(GmaneSource("http://rss.gmane.org/
gmane.comp.lang.r.general"))
```

we get the latest news from the mailing list. **tm** ships with a variety of sources for different purposes, including a source for RSS feeds in the format delivered by Gmane. The source automatically uses a reader function for newsgroups as it detects the RSS feed structure. This means that the meta data in the e-mail headers is extracted, e.g., we obtain

```
> meta(rss[[1]])
Available meta data pairs are:
  Author       : Peter Dalgaard
  Cached       : TRUE
  DateTimeStamp: 2008-04-22 08:16:54
  ID           : http://permalink.gmane.org/
gmane.comp.lang.r.general/112060
  Heading      : R 2.7.0 is released
  Language     : en_US
  Origin       : Gmane Mailing List Archive
  URI          : file http://rss.gmane.org/
gmane.comp.lang.r.general UTF-8
```

for a recent mail announcing the availability of R 2.7.0.

If we want to work with a bigger collection of mails we use the mailing list archives in mailbox format. For demonstration we downloaded the file '2008-January.txt.gz' from `https://stat.ethz.ch/pipermail/r-help/`. In the next step we convert the single file containing all mails to a collection of files each containing a single mail. This is especially useful when adding new files or when lazy loading of the mail corpora is intended, i.e., when the loading of the mail content into memory is deferred until it is first accessed. For the conversion we use the function `convertMboxEml()` which extracts the R-help January 2008 mailing list postings to the directory 'Mails/2008-January/':

```
convertMboxEml(
  gzfile("Mails/2008-January.txt.gz"),
  "Mails/2008-January/")
```

Then we can create the corpus using the created directory. This time we explicitly define the reader to be used as the directory source cannot infer automatically the internal structures of its files:

```
rhelp <- Corpus(DirSource("Mails/2008-January/"),
            list(reader = readNewsgroup))
```

The newly created corpus `rhelp` representing the January archive contains almost 2500 documents:

```
> rhelp
A text document collection with 2486 documents
```

Then we can perform a set of *transformation*s, like converting the e-mails to plain text, strip extra whitespace, convert the mails to lower case, or remove e-mail signatures (i.e., text after a `--␣` (two hyphens, followed by a space) mark).

```
rhelp <- tmMap(rhelp, asPlain)
rhelp <- tmMap(rhelp, stripWhitespace)
rhelp <- tmMap(rhelp, tmTolower)
rhelp <- tmMap(rhelp, removeSignature)
```

This simple preprocessing already enables us to work with the e-mails in a comfortable way, concentrating on the main content in the mails. This way it circumvents the manual handling and parsing of the internals of newsgroup postings.

Since we have the meta data available we can find out who wrote the most postings during January 2008 in the R-help mailing list. We extract the author information from all documents in the corpus and normalize multiple entries (i.e., several lines for the same mail) to a single line:

```
authors <- lapply(rhelp, Author)
authors <- sapply(authors, paste, collapse = " ")
```

Then we can easily find out the most active writers:

```
> sort(table(authors), decreasing = TRUE)[1:3]
Gabor Grothendieck 100
 Prof Brian Ripley  97
    Duncan Murdoch  63
```

Finally we perform a full text search on the corpus. We want to find out the percentage of mails dealing with problems. In detail we search for those documents in the corpus explicitly containing the term `problem`. The function `tmIndex()` filters out those documents and returns their index where the full text search matches.

```
p <- tmIndex(rhelp, FUN = searchFullText,
             "problem", doclevel = TRUE)
```

The return variable `p` is a logical vector of the same size as the corpus indicating for each document whether the search has matched or not. So we obtain

```
> sum(p) / length(rhelp)
[1] 0.2373290
```

as the percentage of explicit problem mails in relation to the whole corpus.

Outlook

The **tm** package provides the basic infrastructure for text mining applications in R. However, there are open challenges for future research: First, larger data sets easily consist of several thousand documents resulting in large term-document matrices. This causes a severe memory problem as soon as dense data structures are computed from sparse term-document matrices. Anyway, in many cases we can significantly reduce the size of a term-document matrix by either removing stopwords, i.e., words with low information entropy, or by using a controlled vocabulary. Both techniques are supported by **tm** via the `stopwords` and `dictionary` arguments for the `TermDocMatrix()` constructor. Second, operations on large corpora are time consuming and should be avoided as far as possible. A first solution resulted in so-called *lazy transformation*s, which materialize operations only when documents are later accessed, but further improvement is necessary. Finally, we have to work on better integration with other packages for natural language processing, e.g., with packages for tag based annotation.

Acknowledgments

I would like to thank Kurt Hornik, David Meyer, and Vince Carey for their constructive comments and feedback.

Bibliography

J. N. G. Binongo. Who wrote the 15th book of Oz? An application of multivariate analysis to authorship attribution. *Chance*, 16(2):9–17, 2003.

I. Feinerer. *tm: Text Mining Package*, 2008. URL `http://CRAN.R-project.org/package=tm`. R package version 0.3-1.

I. Feinerer, K. Hornik, and D. Meyer. Text mining infrastructure in R. *Journal of Statistical Software*, 25(5):1–54, March 2008. ISSN 1548-7660. URL `http://www.jstatsoft.org/v25/i05`.

L. M. Ingebrigtsen. Gmane: A mailing list archive, 2008. URL `http://gmane.org/`.

A. Karatzoglou, A. Smola, K. Hornik, and A. Zeileis. kernlab – an S4 package for kernel methods in R. *Journal of Statistical Software*, 11(9):1–20, 2004. URL `http://www.jstatsoft.org/v11/i09/`.

Ingo Feinerer
Wirtschaftsuniversität Wien, Austria
`feinerer@logic.at`