

This post will go through how to create a word cloud of article titles scraped from the [awesome R-bloggers](https://www.r-bloggers.com/). Our goal will be to use R’s **rvest** package to search through 50 successive pages on the site for article titles. The **stringr** and **tm** packages will be used for string cleaning and for creating a term document frequency matrix (with **tm**). We will then create a word cloud based off the words comprising these titles.

First, we’ll load the packages we need.

# load packages

library(rvest)

library(stringr)

library(tm)

library(wordcloud)

Let’s write a function that will take a webpage as input and return all the scraped article titles.

scrape\_post\_titles <- function(site)

{

# scrape HTML from input site

source\_html <- read\_html(site)

# grab the title attributes from link (anchor) tags within H2 header tags

titles <- source\_html %>% html\_nodes("h2")

%>% html\_nodes("a")

%>% html\_attr("title")

# filter out any titles that are NA (where no title was found)

titles <- titles[!is.na(titles)]

# parse out just the article title (removing the words "Permalink to ")

titles <- gsub("Permalink to ", "", titles)

# return vector of titles

return(titles)

}

The above function takes an input, called *site*, which will be the URL of a specific webpage on [R-bloggers](https://www.r-bloggers.com/). We then use **rvest’s** *read\_html* function to scrape the HTML from the webpage. Next, we parse out the titles by searching through the H2 tags, and parsing out the title attributes from the links within those header tags i.e. we search through each H2 tag, find the “a” tag (anchor, or link), and then pull the title from that tag.

The remaining code above is for cleaning up the titles we parsed. We take out any titles we parsed that are NA – i.e. any link tags that did not have a title attribute (these are not post titles). At this point, each of the title attributes we have has the words “Permalink to “. The *gsub* line of code is just getting rid of this in each title.

# filter out any titles that are NA (where no title was found)

titles <- titles[!is.na(titles)]

# parse out just the article title (removing the words "Permalink to ")

titles <- gsub("Permalink to ", "", titles)

Now, let’s get the vector of webpages we need to scrape. Each successive page containing article links has the following pattern:

**“https://www.r-bloggers.com/page/index”** where index is some positive integer.

<https://www.r-bloggers.com/page/1>  
<https://www.r-bloggers.com/page/2>  
<https://www.r-bloggers.com/page/3>  
<https://www.r-bloggers.com/page/4>  
**…**  
**…**  
**…**

Thus, we can just use the *paste0* function to generate all the URLs we want.

root <- "https://www.r-bloggers.com/"

# get each webpage URL we need

all\_pages <- c(root, paste0(root, "page/", 2:50))

Next, let’s scrape the post titles from each webpage using our *scrape\_post\_titles* function. Then, we’ll collapse the titles into a single vector.

# use our function to scrape the title of each post

all\_titles <- lapply(all\_pages, scrape\_post\_titles)

# collapse the titles into a vector

all\_titles <- unlist(all\_titles)

After we have the titles scraped, we need to perform some cleaning operations, such as converting each title to lowercase, and getting rid of numbers, punctuation, and [stop words](https://en.wikipedia.org/wiki/Stop_words).

## Clean up the titles vector

#############################

# convert all titles to lowercase

cleaned <- tolower(cleaned)

# remove any numbers from the titles

cleaned <- removeNumbers(cleaned)

# remove English stopwords

cleaned <- removeWords(cleaned, stopwords("en"))

# remove punctuation

cleaned <- removePunctuation(cleaned)

# remove spaces at the beginning and end of each title

cleaned <- str\_trim(cleaned)

Next, we use the **tm** package to convert our cleaned vector of titles to a corpus. On the next line, we [stem each word](https://en.wikipedia.org/wiki/Stemming) in the titles to get the root of each word (e.g. model, models, and modeling will each count as the same word, model).

# convert vector of titles to a corpus

cleaned\_corpus <- Corpus(VectorSource(cleaned))

# steam each word in each title

cleaned\_corpus <- tm\_map(cleaned\_corpus, stemDocument)

With the cleaned corpus, we can get a term document matrix. This will give us a frequency of how often each word occurs.

doc\_object <- TermDocumentMatrix(cleaned\_corpus)

doc\_matrix <- as.matrix(doc\_object)

# get counts of each word

counts <- sort(rowSums(doc\_matrix),decreasing=TRUE)

# filter out any words that contain non-letters

counts <- counts[grepl("^[a-z]+$", names(counts))]

# create data frame from word frequency info

frame\_counts <- data.frame(word = names(counts), freq = counts)

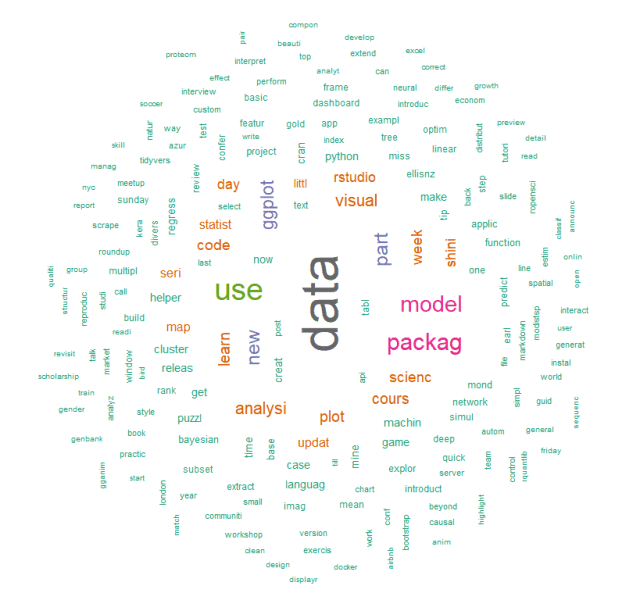
Lastly, we use the **wordcloud** package to generate a word cloud based off the words across all the titles.

set.seed(1000)

wordcloud(words = frame\_counts$word, freq = frame\_counts$freq, min.freq = 1,

max.words=200, random.order=FALSE, rot.per=0.2,

colors=brewer.pal(8, "Dark2"))



Above, we can see that “data” is the most popular word. Variations of “model”, “analysis”, and “package” are also popular.