Lab 07 - Web scraping and Regular Expressions

October 7, 2021

Learning goals

- Use a real world API to make queries and process the data.
- Use regular expressions to parse the information.
- Practice your GitHub skills.

Lab description

In this lab, we will be working with the NCBI API to make queries and extract information using XML and regular expressions. For this lab, we will be using the `httr`, `xml2`, and `stringr` R packages.

This markdown document should be rendered using `github_document` document.

Question 1: How many sars-cov-2 papers?

Build an automatic counter of sars-cov-2 papers using PubMed. You will need to apply XPath as we did during the lecture to extract the number of results returned by PubMed in the following web address:

```
https://pubmed.ncbi.nlm.nih.gov/?term=sars-cov-2
```

Complete the lines of code:

```
# Downloading the website
website <- xml2::read_html("[URL]")

# Finding the counts
counts <- xml2::xml_find_first(website, "[XPath]")

# Turning it into text
counts <- as.character(counts)

# Extracting the data using regex
stringr::str_extract(counts, "[REGEX FOR NUMBERS WITH COMMAS/DOTS]")
```

Don't forget to commit your work!

Question 2: Academic publications on COVID19 and Hawaii

You need to query the following The parameters passed to the query are documented here.

Use the function `httr::GET()` to make the following query:

1. Baseline URL: https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi

2. Query parameters:

   - db: pubmed
   - term: covid19 hawaii
   - retmax: 1000

```
library(httr)
query_ids <- GET(
  url   = "BASELINE URL",
  query = list("QUERY PARAMETERS")
)

# Extracting the content of the response of GET
ids <- httr::content(query_ids)
```

The query will return an XML object, we can turn it into a character list to analyze the text directly with `as.character()`. Another way of processing the data could be using lists with the function `xml2::as_list()`. We will skip the latter for now.

Take a look at the data, and continue with the next question (don't forget to commit and push your results to your GitHub repo!).

Question 3: Get details about the articles

The Ids are wrapped around text in the following way: `<Id>... id number ...</Id>`. we can use a regular expression that extract that information. Fill out the following lines of code:

```
# Turn the result into a character vector
ids <- as.character(ids)

# Find all the ids
ids <- stringr::str_extract_all(ids, "PATTERN")[[1]]

# Remove all the leading and trailing <Id> </Id>. Make use of "|"
ids <- stringr::str_remove_all(ids, "PATTERN")
```

With the ids in hand, we can now try to get the abstracts of the papers. As before, we will need to coerce the contents (results) to a list using:

1. Baseline url: https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi

2. Query parameters:

   - db: pubmed
   - id: A character with all the ids separated by comma, e.g., "1232131,546464,13131"
   - retmax: 1000
   - rettype: abstract

Pro-tip: If you want `GET()` to take some element literal, wrap it around `I()` (as you would do in a formula in R). For example, the text `"123,456"` is replaced with `"123%2C456"`. If you don't want that behavior, you would need to do the following `I("123,456")`.

```
publications <- GET(
  url   = "BASELINE URL HERE",
  query = list(
    "PARAMETERS OF THE QUERY"
    )
)

# Turning the output into character vector
publications <- httr::content(publications)
publications_txt <- as.character(publications)
```

With this in hand, we can now analyze the data. This is also a good time for committing and pushing your work!

Question 4: Distribution of universities, schools, and departments

Using the function `stringr::str_extract_all()` applied on `publications_txt`, capture all the terms of the form:

1. University of …
2. … Institute of …

Write a regular expression that captures all such instances

```
institution <- str_extract_all(
  publications_txt,
  "[YOUR REGULAR EXPRESSION HERE]"
  )
institution <- unlist(institution)
table(institution)
```

Repeat the exercise and this time focus on schools and departments in the form of

1. School of …
2. Department of …

And tabulate the results

```
schools_and_deps <- str_extract_all(
  abstracts_txt,
  "[YOUR REGULAR EXPRESSION HERE]"
  )
table(schools_and_deps)
```

Question 5: Form a database

We want to build a dataset which includes the title and the abstract of the paper. The title of all records is enclosed by the HTML tag `ArticleTitle`, and the abstract by `Abstract`.

Before applying the functions to extract text directly, it will help to process the XML a bit. We will use the `xml2::xml_children()` function to keep one element per id. This way, if a paper is missing the abstract, or something else, we will be able to properly match PUBMED IDS with their corresponding records.

```
pub_char_list <- xml2::xml_children(publications)
pub_char_list <- sapply(pub_char_list, as.character)
```

Now, extract the abstract and article title for each one of the elements of `pub_char_list`. You can either use `sapply()` as we just did, or simply take advantage of vectorization of `stringr::str_extract`

```
abstracts <- str_extract(pub_char_list, "[YOUR REGULAR EXPRESSION]")
abstracts <- str_remove_all(abstracts, "[CLEAN ALL THE HTML TAGS]")
abstracts <- str_remove_all(abstracts, "[CLEAN ALL EXTRA WHITE SPACE AND NEW LINES]")
```

How many of these don't have an abstract? Now, the title

```
titles <- str_extract(pub_char_list, "[YOUR REGULAR EXPRESSION]")
titles <- str_remove_all(titles, "[CLEAN ALL THE HTML TAGS]")
```

Finally, put everything together into a single `data.frame` and use `knitr::kable` to print the results

```
database <- data.frame(
  "[DATA TO CONCATENATE]"
)
knitr::kable(database)
```

Done! Knit the document, commit, and push.

Final Pro Tip (optional)

You can still share the HTML document on github. You can include a link in your `README.md` file as the following:

```
View [here](https://ghcdn.rawgit.org/:user/:repo/:tag/:file)
```

For example, if we wanted to add a direct link the HTML page of lecture 7, we could do something like the following:

```
View [here](https://ghcdn.rawgit.org/USCbiostats/PM566/master/website/static/slides/07-apis-regex/slides.html)
```

PM566: Introduction to Health Data Science - PM 566 (Fall 2021)

University of Southern California

Department of Population and Public Health Sciences

George Vega Yon, Kim Siegmund, Abigail Horn

vegayon@usc.edu