You might have heard of the [Advent of Code](https://adventofcode.com/),  
a 25-day challenge involving a programming puzzle a day, to be solved  
with the language of your choice. I’ve noted the popularity of this  
activity in my Twitter timeline but also in my GitHub timeline where  
I’ve seen the creation of a few advent-of-code or so repositories.

AoC is largely an exercise in figuring how to write your favourite language as if were C or C++ , which can be fun … in moderation

— Jenny Bryan (@JennyBryan) [December 12, 2018](https://twitter.com/JennyBryan/status/1072724282705203201?ref_src=twsrc%5Etfw)

If I were to participate one year, I’d probably use R. Jenny Bryan’s  
tweet above inspired me to try and gauge the popularity of languages  
used in the Advent of Code. To do that, in this post, I shall use the  
search endpoint of GitHub V3 API to identify Advent of Code 2018 repos.

**Searching repositories on GitHub**

**Study design **

GitHub’s V3 API offers a [search  
endpoint](https://developer.github.com/v3/search/#search-repositories),  
that however gives you less results than doing the [same search via the  
web interface](https://github.com/search?q=adventofcode+2018), even when  
using pagination right (or at least, when believing I use pagination  
right!). I’m however willing to use that sub-sample as basis for my  
study of language popularity. It’s actually a sub-sub-sample, since I’m  
only looking at Advent of code projects published on GitHub.

In order to circumvent the sub-sub-sampling a bit, I’ll do the search in  
two steps:

* Searching for Advent of code 2018 in general among repos, and  
  extracting the language of the repos.
* Searching for Advent of code 2018 *for each of these languages  
  separately* and extracting the total count of hits.

Note that I am not filtering the repos by activity, so some of them  
could very well have been created for a few days only. If they are empty  
though, they do not get assigned a language.

Regarding the language of repos, GitHub assigns a language to each  
repository. This information can be wrong, which is e.g. mentioned in  
[rOpenSci’s development  
guide](https://ropensci.github.io/dev_guide/grooming.html#github-linguist).  
Furthermore, my using this piece of information means I’m disregarding  
the fact that [some people actually use a mix of technologies to solve  
the puzzles](https://www.benkraft.org/2017/12/26/advent-of-code/).

**Actual queries**

I first defined a function to search the API whilst respecting the rate  
limiting. I even erred on the side of caution and queried very slowly.

.search <- function(page){

gh::gh("GET /search/repositories",

q = "adventofcode 2018",

page = page,

fork = FALSE)

}

search <- ratelimitr::limit\_rate(.search,

ratelimitr::rate(10, 60))

I then wrote two other functions to help me rectangle the API output for  
each repository.

empty\_null <- function(x){

if(is.null(x)){

""

}else{

x

}

}

rectangle <- function(item){

tibble::tibble(full\_name = item$full\_name,

language = empty\_null(item$language))

}

I created a function putting these two pieces together.

get\_page <- function(page){

results <- try(search(page), silent = TRUE)

# an early return

if(inherits(results, "try-error")){

return(NULL)

}

purrr::map\_df(results$items,

rectangle)

}

And I then ran the following pipeline.

total\_count <- search(1)$total\_count

pages <- 1:(ceiling(total\_count/100))

results <- purrr::map\_df(pages, get\_page)

results <- unique(results)

languages <- unique(results$language)

languages <- languages[languages != ""]

This got me 814 repos, with 46 non empty languages. Repo names are quite  
varied: rdmueller/aoc-2018, petertseng/adventofcode-rb-2018,  
NiXXeD/adventofcode, Arxcis/adventofcode2018,  
Stupremee/adventofcode-2018, phaazon/advent-of-code-2k18.

With that information obtained, I was able to run a query by language.

.get\_one\_language\_count <- function(language){

gh::gh("GET /search/repositories",

q = glue::glue("adventofcode 2018&language:{language}"),

fork = FALSE)$total\_count -> count

tibble::tibble(language = language,

count = count)

}

get\_one\_language\_count <- ratelimitr::limit\_rate(.get\_one\_language\_count,

ratelimitr::rate(10, 60))

counts <- purrr::map\_df(languages,

get\_one\_language\_count)

In total, the counts table contains information about 2080  
repositories, a bit less than half the number of Advent of code 2018  
repositories I’d find via the web interface.

**Advent of Code’s languages popularity**

I’ll concentrate on the 15 most popular languages in the sample, which  
automatically excludes R with… 8 repositories only.

library("ggplot2")

library("ggalt")

library("hrbrthemes")

library("magrittr")

counts %>%

dplyr::arrange(- count) %>%

head(n = 15) %>%

dplyr::mutate(language = reorder(language, count)) %>%

ggplot() +

geom\_lollipop(aes(language, count),

size = 2, col = "salmon") +

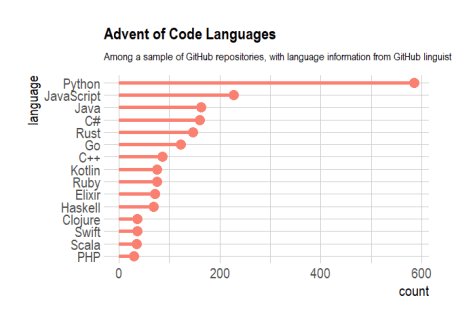
hrbrthemes::theme\_ipsum(base\_size = 16,

axis\_title\_size = 16) +

coord\_flip() +

ggtitle("Advent of Code Languages",

subtitle = "Among a sample of GitHub repositories, with language information from GitHub linguist")



The results are not surprising after reading e.g. the [insights from  
Stack Overflow’s 2018  
survey](https://insights.stackoverflow.com/survey/2018/#technology), although the Python domination is crazy! I find interesting to reflect on the  
fact that Jenny Bryan says that the challenge is best for C or C++, that  
are not the most popular languages in these samples… but still more  
popular than R, ok.

**Conclusion**

In this post I used GitHub V3 API to get a glimpse at the popularity of languages used to solve the Advent of Code. Further work could include looking at the *completion* of the challenge by language, potentially using the GitHub activity of each repo as an (imperfect) proxy.

I do not take part in the challenge myself, my principal Advent’s  
specific activity being instead the lazy and delightful watching of the  
[Swedish TV channel SVT Adventskalender for  
kids](https://sv.wikipedia.org/wiki/Julkalendern_i_Sveriges_Television).  
Incidentally, [this year’s  
storyline](https://sv.wikipedia.org/wiki/Storm_p%C3%A5_Lugna_gatan)  
includes a Christmas competition, which however features competitive  
eating of saffron buns and gingerhouse building rather than programming  
puzzles… Do *you* participate to Advent of Code this year? If so, with  
which language and why?