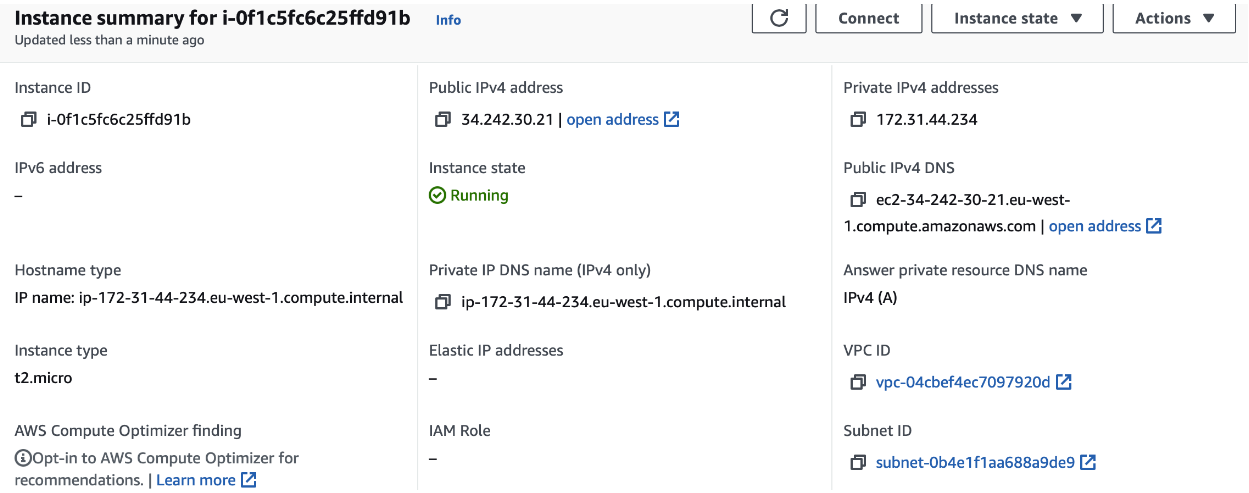
Assignment 3

2022-04-18

I created an account in AWS and connect R server by using Public IPv4 address: 34.242.30.21. However, I am not able to use spark on R server. Therefore, as mentioned in the assignment, I download spark and java to use locally. The screenshot below shows the AWS information.



Instance summary for i-0f1c5fc6c25ffd91b

Updated less than a minute ago

Instance ID
i-0f1c5fc6c25ffd91b

Public IPv4 address
34.242.30.21 | open address

Private IPv4 addresses
172.31.44.234

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-34-242-30-21.eu-west-1.compute.amazonaws.com | open address

Hostname type
IP name: ip-172-31-44-234.eu-west-1.compute.internal

Private IP DNS name (IPv4 only)
ip-172-31-44-234.eu-west-1.compute.internal

Answer private resource DNS name
IPv4 (A)

Instance type
t2.micro

Elastic IP addresses
–

VPC ID
vpc-04cbef4ec7097920d

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

IAM Role
–

Subnet ID
subnet-0b4e1f1aa688a9de9

```r
rm(list = ls())
library(sparklyr)
library(tidyverse)
library(ggplot2)
utils::sessionInfo()
```

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS  10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] forcats_0.5.1   stringr_1.4.0   dplyr_1.0.7     purrr_0.3.4
##  [5] readr_1.4.0     tidyr_1.2.0     tibble_3.1.2    ggplot2_3.3.5
##  [9] tidyverse_1.3.1 sparklyr_1.7.5
##
## loaded via a namespace (and not attached):
```

```
##  [1] tidyselect_1.1.1  xfun_0.23         forge_0.2.0       haven_2.4.1      
##  [5] colorspace_2.0-1  vctrs_0.3.8       generics_0.1.0    htmltools_0.5.1.1
##  [9] yaml_2.2.1        base64enc_0.1-3   utf8_1.2.1        rlang_0.4.11     
## [13] pillar_1.6.1      withr_2.4.2       glue_1.4.2        DBI_1.1.1        
## [17] dbplyr_2.1.1      modelr_0.1.8      readxl_1.3.1      lifecycle_1.0.0  
## [21] munsell_0.5.0     gtable_0.3.0      cellranger_1.1.0  rvest_1.0.0      
## [25] htmlwidgets_1.5.4 evaluate_0.14     knitr_1.33        fansi_0.5.0      
## [29] broom_0.7.8       r2d3_0.2.6        Rcpp_1.0.7        backports_1.2.1  
## [33] scales_1.1.1      jsonlite_1.7.2    fs_1.5.0          hms_1.1.0        
## [37] digest_0.6.27     stringi_1.6.2     rprojroot_2.0.2   grid_3.6.1       
## [41] cli_3.0.0         tools_3.6.1       magrittr_2.0.1    crayon_1.4.1     
## [45] pkgconfig_2.0.3   ellipsis_0.3.2    xml2_1.3.2        reprex_2.0.0     
## [49] lubridate_1.7.10  assertthat_0.2.1  rmarkdown_2.8     httr_1.4.2       
## [53] rstudioapi_0.13   R6_2.5.0          compiler_3.6.1
```

In this project, we need to use spark to analyze data, here I connect to saprk and initialize it.

```r
# spark_install()
# when first
#spark_install(version = "3.2.1")
sc <- spark_connect(master = "local")
```

Here, I read the data and then rename the irregular variable names. Then add the two datasets to saprk. The two data are then merged in spark, and the data is simply cleaned as needed.

```r
# import data
datafips=read.csv("UID_ISO_FIPS_LookUp_Table.csv")
dataglb=read.csv("time_series_covid19_confirmed_global.csv")
# rename
dataglb <- rename(dataglb, c(Province_State = Province.State,
                             Country_Region = Country.Region))
datafips <- rename(datafips, Long = Long_)
# add the two datasets
dataglb_tbl <- copy_to(sc, dataglb, "dataglb")
datafips_tbl <- copy_to(sc, datafips, "datafips")

# merge the data
datajoin_tbl <- inner_join(dataglb_tbl, datafips_tbl,
                       by = c("Province_State", "Country_Region", "Lat", 'Long')) %>%
  filter(Country_Region %in% c('Germany', 'China', 'Japan', 'United Kingdom',
                               'US', 'Brazil', 'Mexico')) %>%
  pivot_longer(starts_with('X'), names_to = 'Date',  values_to = 'Cases') %>%
  mutate(Date = regexp_replace(Date, 'X', ''),
         Date = regexp_replace(Date, '_', '.'))
# datajoin_tbl$Date = as.Date(datajoin_tbl$Date, "%m_%d_%y")
# summarise
datajoin_tbl2 <- datajoin_tbl %>%
  group_by(Country_Region, Date) %>%
  summarise(Case = sum(Cases, na.rm = T),
            Population = sum(Population, na.rm = T)) %>%
  mutate(Rate = Case/Population*100000) %>%
  filter(Case > 0) %>%
  ungroup()
```

From the figure, it can be found that each country has a different growth trend of cumulative infections from January 2020 to January 2022. It can be seen from the figure that the country with the first cases is China, but the cumulative cases in China began to grow slowly at the end of March 2020, and began to grow rapidly in March 2022. For the rest of the countries, it started to grow rapidly in April 2020, and the growth rate has been maintained at a high level since then. From January 2021, among these countries, the United States will have the most infections and China the least.

```
ggplot(datajoin_tbl2, aes(x = as.Date(Date, "%m.%d.%y"), y = Case, col = Country_Region)) +
  geom_line() +
  theme_bw() +
  theme(legend.position = 'top') +
  labs(x = '') +
  scale_y_log10()
```

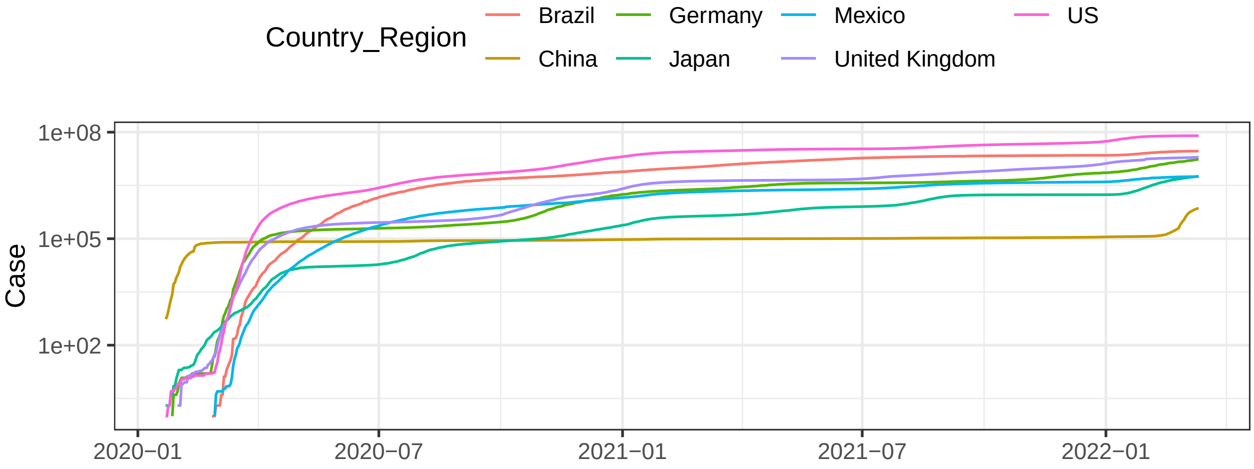


Figure 1: Cumulative infections for COVID-19

Here, I plot the cumulative infection rate per 100,000 people in these countries. It can be seen from the figure that the United States has the highest infection rate until November 2021, after which the cumulative infection rate of the United Kingdom overtakes the United States. In late October 2021, the cumulative infection rate in Germany also showed a trend of increasing significantly. In contrast, China's cumulative infection rate has been the smallest. And it can be found in the middle picture that as of February 2022, the cumulative infection rate in the United Kingdom, the United States and Germany is 20,000 per 100,000 people, which means that nearly two in 10 people are infected with COVID-19.

```
ggplot(datajoin_tbl2, aes(x = as.Date(Date, "%m.%d.%y"), y = Rate, col = Country_Region)) +
  geom_line() +
  theme_bw() +
  theme(legend.position = 'top') +
  labs(x = '',y = 'Rate (per 100,000 people)')
```

The coefficient of Days is 0.0032, which shows that the cumulative number of people infected with COIVD is increasing by 0.32% per day. The coefficient of `log(Population)` is 1.11, which means that for every 1% increase in the population of an area, its cumulative number of infections will increase by about 1.11%. The coefficient of `Country_RegionChina` is -6.81, which implies that the cumulative number of infections in China is about 682% lower than in Brazil. The coefficient of `Country_RegionGermany` is -0.62, which implies that the cumulative number of infections in Germany is about 62% lower than in Brazil. The coefficient of `Country_RegionJapan` is -2.66, which implies that the cumulative number of infections in Japan is about 266% lower than in Brazil. The coefficient of `Country_RegionMexico` is -1.12, which implies that the cumulative number of infections in Mexico is about 112% lower than in Brazil. The coefficient of `Country_RegionUnited Kingdom` is 0.24, which implies that the cumulative number of infections in the UK is about 24% higher than in Brazil. The coefficient of `Country_RegionUS` is -0.07, which implies that the cumulative number of infections in the US is about 7% lower than in Brazil.

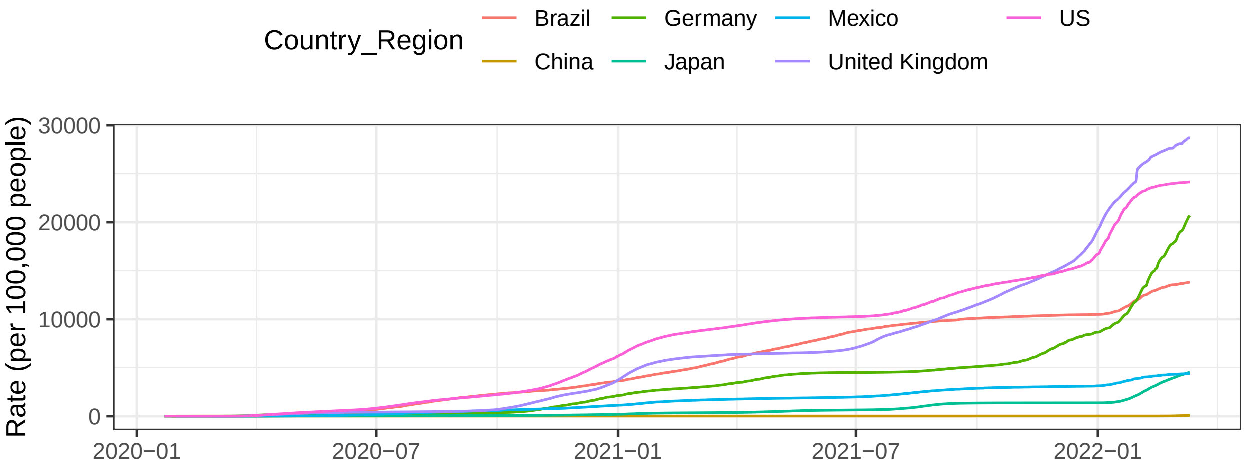


Figure 2: Cumulative infections (per 100,000 people)

```
datajoin_tbl3 <- datajoin_tbl %>%
  mutate(lcase = log(Cases),  # calculate the log
         Date = to_date(Date, 'm.d.yy'),
         Days = datediff(Date, '2019-12-31')) %>%
  select(lcase, Days, Country_Region, Population)

mod <-  lm(lcase ~ Country_Region + log(Population) + Days, data = datajoin_tbl3)
coef(mod)
```

```
##                  (Intercept)          Country_RegionChina
##                 -6.972141783                 -6.816704394
##        Country_RegionGermany          Country_RegionJapan
##                 -0.616068334                 -2.663943221
##         Country_RegionMexico Country_RegionUnited Kingdom
##                 -1.122782226                  0.235439007
##             Country_RegionUS              log(Population)
##                 -0.074404904                  1.112656858
##                         Days
##                  0.003184034
```

I execute the following error, so I choose lm function.

```
mod <- datajoin_tbl3 %>%
  ml_linear_regression(lcase~.)
```