

PILAS Tutorial

Sofía Meléndez Cartagena

7/20/2021

Introducción

Paquetes necesarios

- devtools
 - Paquete utilizado para bajar y crear paquetes fuera de CRAN (el repositorio oficial de R)
- PilaPack
 - Paquete generado por el equipo de trabajo de PILAS se puede bajar aquí:
 - `devtools::install_github("ComplejoC/PILASPack/PilasPack")`
- Tidyverse
 - Necesario para graficar y crear un proceso de conductos en R
- Lubridate
 - Utilizado para manipular fechas
- httr
 - Utilizado para interactuar con el API de Bioportal
- jsonlite
 - Utilizado para transformar los tokens del API

Adquisición de datos

Estamos en proceso de arreglar la función de adquisición de datos en el paquete, asta nuevo aviso se utiliza este código:

```
## Download Prep

start_of_month=list("01/01/2021", "02/01/2021", "03/01/2021", "04/01/2021", "05/01/2021")
names(start_of_month)=c("January2021", "February2021", "March2021", "April2021", "May2021")

end_of_month=list("01/31/2021", "02/28/2021", "03/31/2021", "04/30/2021", "05/31/2021")
names(end_of_month)=c("January2021", "February2021", "March2021", "April2021", "May2021")

paste("/?", "startDate=", start_of_month[4], "&", "endDate=", end_of_month[4], sep="")
## Download Data
```

```

l=length(start_of_month)

base_url="https://bioportal.salud.gov.pr/api/administration/reports/unique-tests"
Datos_all_list=list()

for (i in 1:l){

  print(Sys.time())

  #produce the url
  url_req=paste("/?", "createdAtStartDate=", start_of_month[i], "&",
                "createdAtEndDate=", end_of_month[i], "sep=")
  url=paste(base_url, url_req, sep="")

  data <- list(
    email="",
    password = "",
    reCaptchaResponseToken="N/A")

  res = POST("https://bioportal.salud.gov.pr/api/authentication/login/",
             body= data, encode='json', content_type('application/json'))

  token = fromJSON(rawToChar(res$content))

  datos = GET(url,
              content_type('application/json'),
              add_headers(authorization = paste('Bearer', token$token),
                          'Accept-Enconding'="br"))

  datos = fromJSON(rawToChar(datos$content), flatten = TRUE)

  ## Copias de base de datos ##
  Datos<-datos

  #store the data in a list

  Datos_all_list[[i]]<-Datos

  names(Datos_all_list)<-names(start_of_month)[c(1:i)]

  print(names(start_of_month[i]))
}

## Concatenate
Datos_all<-do.call("rbind", Datos_all_list)

```

Hagase ajuste de los meses de interes así como sea necesario para su analisis. El espacio para email y password deben ser rellenos con su informacion de bioportal.

Preprocesamiento

Utilizando las funciones de PilasPack se va transformar las variables de fechas a formato de fecha y además se va calcular los rezagos, que en nuestra tabla se llamarán:

- TatReportSample
 - La diferencia entre la fecha que el resultado es reportado en el laboratorio y la fecha en la que se tomó la muestra
- TatUploadReport
 - La diferencia entre la fecha que el resultado es reportado al bioportal y el resultado reportado en el laboratorio
- TatUploadSample
 - La diferencia entre la fecha que el resultado es reportado al bioportal y la fecha en la que se tomó la muestra

Tat es corto para Turn Around Time.

Ejemplares del preproceso donde Datos_all son los datos tal cual fueron bajados de Bioportal:

```
All_test <- testApiToDates(DataFrame = Datos_all)
head(All_test)
```

Esta función anterior transforma las columnas de fechas al formato de fecha

```
All_test <- testApiCalculateTurnaround(All_test)
head(All_test)
```

Esta función anterior calcula los rezagos y añade tres columnas a la tabla con esos valores

```
All_test <- testApiMonthAsColumn(All_test)
head(All_test)
```

Esta función anterior extrae el mes en el que las diferentes fechas claves ocurrieron y las añade al final de la tabla. Esta última función es opcional en el proceso.

Errores comunes y que hacer con ellos

El Bioportal empezó a recibir datos el “2020-03-9”, se supone que ya se alló arreglado el error que llevo a que aparecieran pruebas más viejas que esa fecha. Sin embargo siempre es una buena práctica corroborar. En una tabla aparte aísla los datos con fecha errónea para enviársela a la oficina de Bioportal, en la tabla principal elimina los datos erróneos.

```
DataUpBeforeMarch <- All_test %>%
  filter(createdAtDate < ymd("2020-03-09"))
```

```
All_test <- All_test %>%
  filter(createdAtDate > ymd("2020-03-09"))
```

Tambien no es muy fuera de lo comun que las pruebas sean muestreadas en fechas imposibles, recomiendo eliminar cualquier prueba tomada antes de “2020-02-01”. Al igual que la vez anterior, separamos los datos con fechas muy viejas en una tabla aparte y manten el resto de los datos en la tabla original.

```
SampledUpBeforeFeb <- All_test %>%
  filter(sampleCollectedDate < ymd("2020-02-01"))
```

```
All_test <- All_test %>%
  filter(sampleCollectedDate > ymd("2020-02-01"))
```

Otros errores comunes en fechas se tienden a reflejar en los rezagos. Muchas veces encontramos que tenemos rezagos negativos por que la fecha de muestreo es despues de cuando se reporto o se subio a bioportal. Igualmente ocurre que la fecha de reporte es luego de la fecha en que se sube a bioportal. Estos errores tabien los aislamos y los reportamos, pero tienen un protocolo un poco diferente. Primero se aísla todas las pruebas con Tats negativo en una misma tabla y luego se corrobora por duplicados.

```
NegTat <- All_test %>%
  filter(as.numeric(TatReportSample) < 0)%>%
  filter(as.numeric(TatUploadReport) < 0)%>%
  filter(as.numeric(TatUploadSample) < 0)
```

```
NegTat <- NegTat[!duplicated(NegTat),]
```

```
All_test <- All_test %>%
  filter(as.numeric(TatReportSample) > 0)%>%
  filter(as.numeric(TatUploadReport) > 0)%>%
  filter(as.numeric(TatUploadSample) > 0)
```

Analisis comunes

Volumen

```
All_Tests %>%
  select(sampleCollectedDate, TatUploadSample)%>%
  mutate(sampleCollectedDate = month(sampleCollectedDate, label = TRUE)) %>%
  group_by(sampleCollectedDate)%>%
  mutate(TatUploadSample = as.numeric(TatUploadSample))%>%
  summarise(meanTat = mean(TatUploadSample), SDETat = sd(TatUploadSample))%>%
  ggplot(aes(x = sampleCollectedDate, y = meanTat))+
  geom_point()+
  geom_errorbar(aes(ymin=meanTat-SDETat, ymax=meanTat+SDETat))+
  labs(y = "Rezago Promedio", x = "Mes",
       title = "Promedio y Desviacion Estandar en Rezago Entre Muestreo y Subir")+
  theme_bw()
```

Volumen promedio global

Rezago

Campos Vacios

Relacion laboratorio-laboratorio