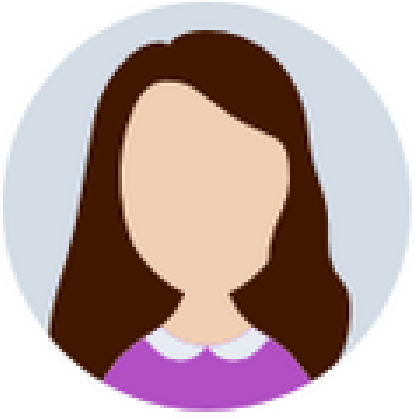
R aplicado al análisis cualitativo / FCS-UdelaR

Clase 3 - Educación Permanente FCS



Mag. Elina Gómez (UMAD/FCS)

elina.gomez@cienciassociales.edu.uy

www.elinagomez.com

Objetivos de hoy

- Fuentes de datos: *rtweet*
- Procesamiento de strings

Credenciales

1. Creamos la App en Twitter Dev
2. Copiamos el bearer token
3. Corremos la función rtweet_app() asignada a un objeto
4. Guardamos con la función auth_save()
5. Cada vez cargamos con auth_as()

Credenciales

```
auth <- rtweet_app(bearer token)
auth_save(auth,"authRCuali")
auth_as("authRCuali")
```

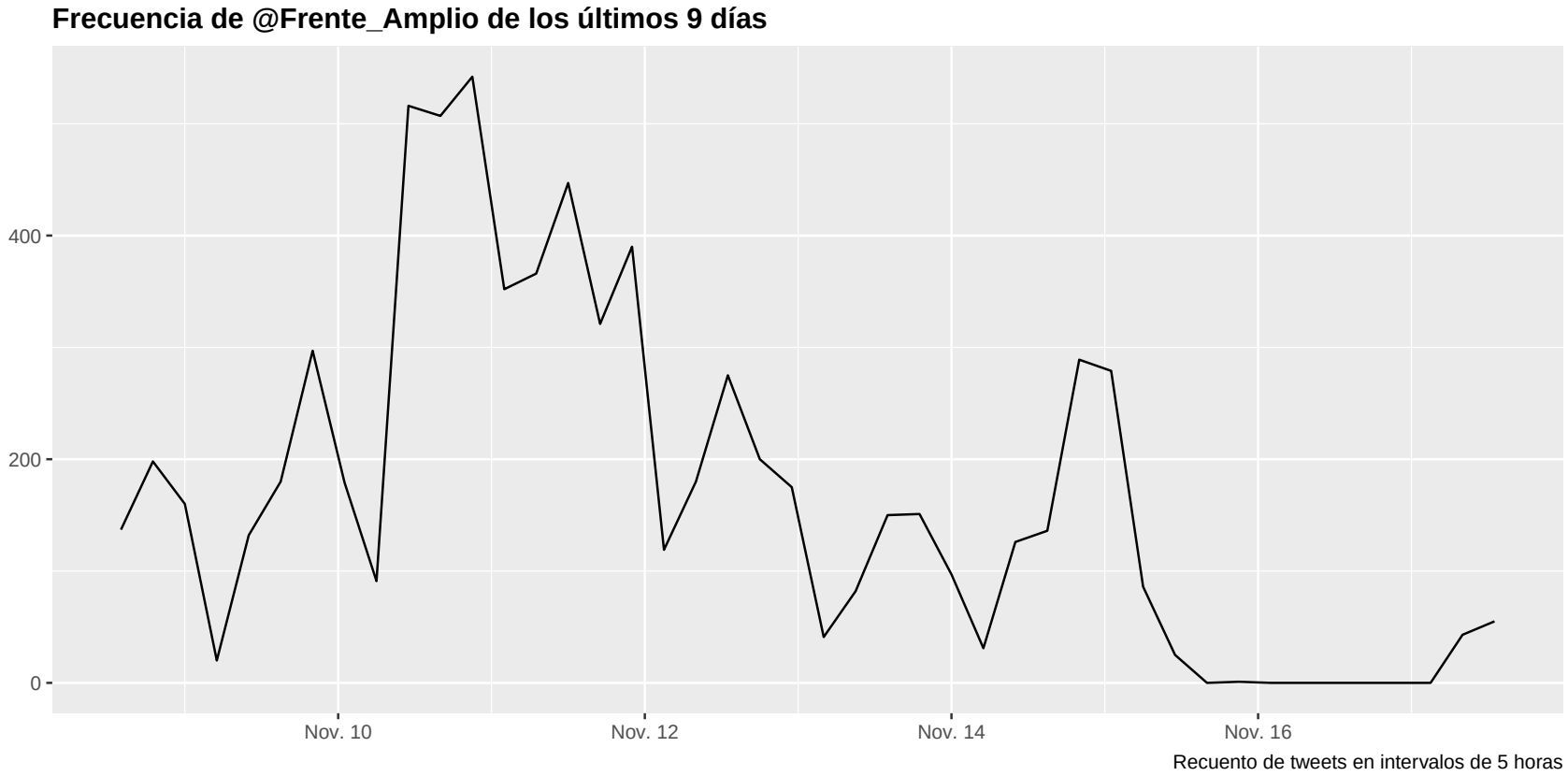
search_tweets()

```
fa <- search_tweets("@Frente_amplio", n = 18000, include_rts = FALSE)

#Si quiero más límite:
#retryonratelimit=TRUE
```

search_tweets()

Los grafico usando la función *ts_plot* para hacer series de tiempo.



search_tweets()

Los grafico usando la función *ts_plot* para hacer series de tiempo.

```
fa %>%
ts_plot("5 hours") +
ggplot2::theme(plot.title = ggplot2::element_text(face = "bold")) +
ggplot2::labs(x = NULL, y = NULL,
title = "Frecuencia de @Frente_Amplio de los últimos 9 días",
caption = "Recuento de tweets en intervalos de 5 horas")
```

get_followers()

Esta función nos da un data.frame con el id de lxs usuarixs. Es importante poner el argumento $n = Inf$ para que incluya a todxs lxs usuarixs.

```
fa_flw = get_followers("Frente_Amplio", n = Inf)
#también retryonratelimit=TRUE para > 75000

#para saber a quién sigue, sería:
fa_frnd = get_friends("Frente_Amplio", n = Inf)
```

lookup_users()

La función *lookup_users()* nos permite obtener información de esos usuarixs. El primer argumento de la función es el nombre del objeto que contiene los ids de lxs seguidorxs.

```
fa_flw_data = lookup_users(fa_flw$from_id)
```

lookup_users()

Con esta función podemos procesar información de diversa índole como:

- Cantidad de seguidorxs
- Cantidad de Tweets (incluidos los retweets) emitidos por el usuario
- La fecha que la cuenta de usuario fue creada en Twitter
- Localización

Para ver todas las variables que incluye:

Todas las variables que incluye

Cantidad de seguidorxs

```
library(RColorBrewer)
require(forcats)
library(ggplot2)

ggplot(seguidores, aes(fct_infreq(candi)))+ geom_bar(fill= c("#FFB5E8","85E3FF")) +
  geom_text(stat='count', aes(label=..count..), vjust=-1)+
   xlab("Politicxs") +
  ylab("Cantidad de seguidorxs")
```

get_favorites()

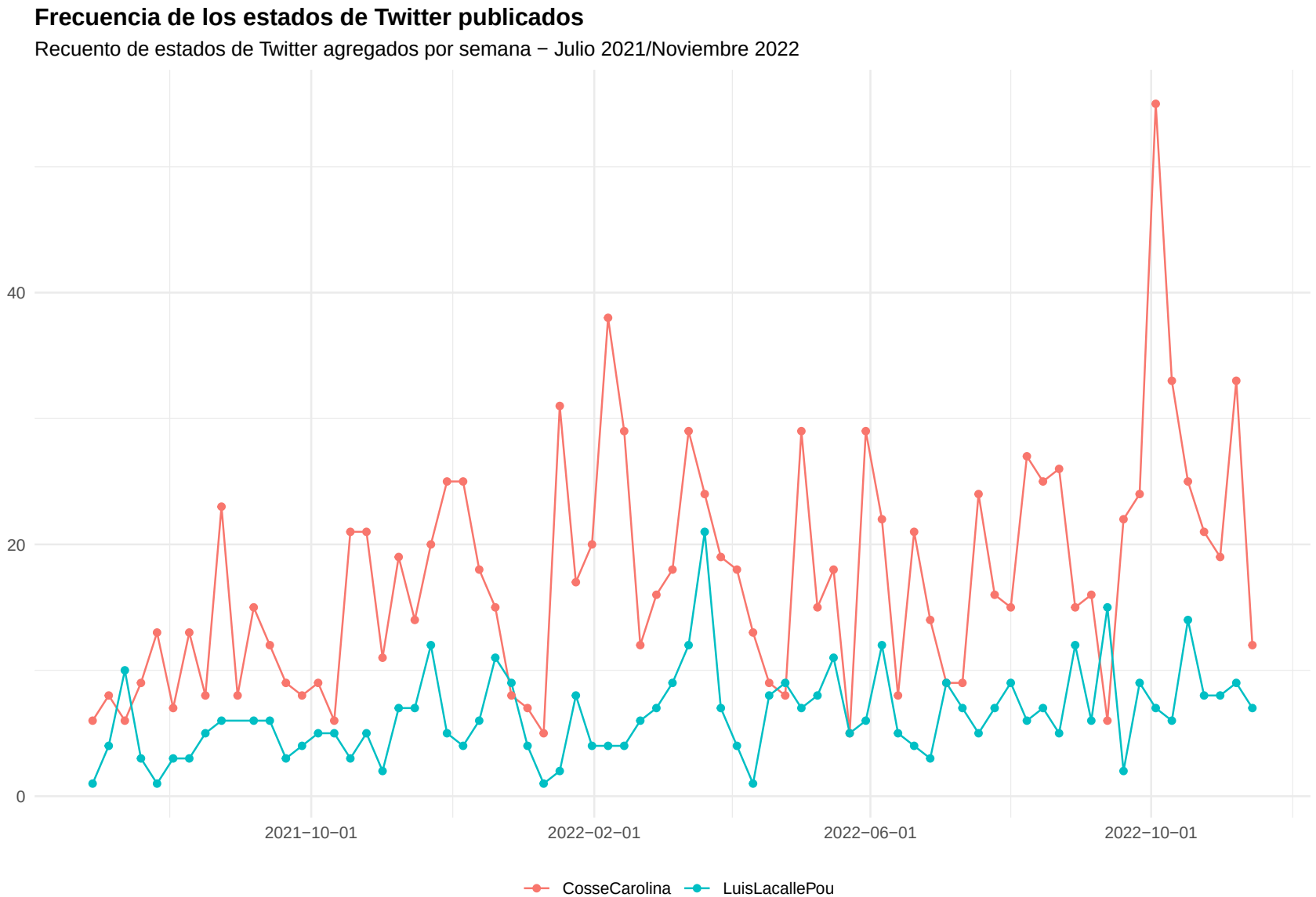
Obtengo los *n* estados favoritos más recientes de unx usuarix.

```
dm_fav = get_favorites("Frente_Amplio", n = 2000)
```

get_timelines()

Obtengo los últimos 3200 estados de Twitter de unx o más usuarixs, para graficar series de tiempo con su frecuencia, según periodicidad de interés.

get_timelines()



Distribución espacial

Estas consultas también pueden ser trabajadas por lo menos a nivel orientativo espacialmente, pudiendose obtener diversos mapeos

Referencias:

- https://rpubs.com/camilamila/tweets
- https://github.com/mkearney/rtweet
- https://mkearney.github.io/nicar_tworkshop/#1

Procesamiento de *strings*

Antes de realizar un análisis o de construir un modelo de aprendizaje, la discusión de datos es un paso crítico para preparar los datos de texto sin procesar en un formato apropiado.

El texto puede ser considerado como una colección de documentos y un documento puede ser analizado en cadenas. En la limpieza de texto, los patrones de búsqueda se definen en expresiones regulares (abreviadas como regex() o regexp()) para "encontrar y eliminar" o "buscar y reemplazar" cadenas.

grep

grep(patrón, cadena) devuelve por defecto una lista de índices. Si la expresión regular, patrón, coincide con un elemento particular en la cadena vectorial, devuelve el índice del elemento.

Para devolver los valores reales de los elementos coincidentes, establezca la opción value=TRUE.

- Creamos un vector "strings" y asignamos cuatro valores (string) a la variable. strings <- c("abcd", "cdab", "cabd", "c abd")

grep

- Buscamos los valores que contengan 'ab', y nos indica sus posiciones grep("ab", strings)
- Que es lo mismo que... grep("ab", strings, value = FALSE)

-Mientras que para obtener los valores usaremos... grep("ab", strings, value = TRUE)

sub() y gsub()

gsub(patrón, remplazo, string) devuelve la cadena modificada después de reemplazar cada ocurrencia de patrón con reemplazo en cadena.

sub(patrón, remplazo, string) reemplaza la primera aparición del patrón.

Stringr

Funciones para separar un texto:

```
library(stringr)

str_split(x, "\r")

str_split(x, boundary("word"))

##type = c("character", "line_break", "sentence", "word")
```

Stringr

Funciones para combinar un texto:

```
library(stringr)

str_c("x", "y", sep = ", ")
> [1] "x, y"
##sep para controlar
```

Stringr

Funciones para reemplazar un texto:

```
library(stringr)
#la primer coincidencia
str_replace(string, pattern, replacement)

#todas las coincidencias
str_replace_all(string, pattern, replacement)
```

Stringr

Funciones para pasar mayúscula/minúscula:

```
library(stringr)

str_to_upper(c("i", "ı"))
#> [1] "I" "I"

str_to_lower(c("I", "İ"))
#> [1] "i" "ı"
```

Stringr

Funciones para pasar eliminar espacios en blanco:

```
library(stringr)

str_trim(string, side = c("both", "left", "right"))

str_trim(" String with trailing and leading white space\t")
#> [1] "String with trailing and leading white space"
str_trim("\n\nString with trailing and leading white space\n\n")
#> [1] "String with trailing and leading white space"
```

Stringr

Funciones para pasar eliminar espacios en blanco:

```
library(stringr)

str_squish(" String with trailing, middle,   and leading white space\t")
#> [1] "String with trailing, middle, and leading white space"
str_squish("\n\nString with excess, trailing and leading white space\n\n")
#> [1] "String with excess, trailing and leading white space"
```

Caracteres especiales

Para construir consultas que incluyan metacaracteres, i.e.

`\\$ \\* \\+ \\. \\? \\[ \\] \\^ \\{ \\} \\| \\( \\)`

Se debe agregar una retrobarra `\\`

Metacaracteres especiales

`\\t` : Tabulador

`\\n` : Nueva línea

`\\v` : Tabulación vertical

`\\f` : Salto de formulario

`\\r` : Salto de línea