Manejo de bases de datos

Buenas prácticas de la programación.

Stalyn Guerrero

Universidad el Bosque

2022-06-17

Section 1

Introducción

Operadores lógicos

- Operadores de relación: `<`, `>`, `<=`, `>=`, `==`, `!=`.
- Operadores lógicos : `!`, `&`, `|`, `&&`, `||`, `xor()`.
- Coincidencia de valores : `%in%`.

Nota:

Los operadores `&&` y `||` siempre solo devuelve un solo `TRUE`o `FALSE`. Estas expresiones evalúan sus argumentos de izquierda a derecha, examinando solo el primer elemento de cada vector. La evaluación procede sólo hasta que se determina el resultado. Por ejemplo, si el primer argumento en una `&&`comparación es `FALSE`el segundo argumento no se evalúa porque ya está claro que el resultado será `FALSE`.

Pruebas lógicas

Además de estos operadores, existen una serie de funciones útiles que se pueden utilizar en ejecución condicional. Ya hemos visto algunos de ellos en los capítulos anteriores.

- `all()`: ¿Todos son elementos `TRUE`?
- `any()`: ¿Es al menos un elemento `TRUE`?
- `all.equal()`: ¿Son dos objetos (casi) iguales?

Estructuras de control

- if

```
x <- 8
if (x < 10) { cat("x es menos que 10\n") }
```

```
## x es menos que 10
```

- if y else

```
x <- 8
if (x < 10) {
    print("x es menos que 10")
} else {
    print("x es mayor o igual a  10")
}
```

```
## [1] "x es menos que 10"
```

Creando loop y estructuras de control

Los bucles son uno de los elementos básicos de todos los lenguajes de programación, no solo `R`, y pueden ser una herramienta poderosa

- `for`
- `while`
- `replicate`

¿Cuándo usar un bucle?

Los bucles se implementan de manera ineficiente en `R` y deben evitarse cuando existen mejores alternativas, especialmente cuando se trabaja con grandes conjuntos de datos. Sin embargo, los bucles son a veces la única forma de lograr el resultado que queremos.

Ejemplo

- Simulaciones
- Relaciones recursivas (*gibbs*)
- Problemas más complejos

Si no hay bucles, ¿entonces qué?

En resumen, utilice la familia de funciones `apply`; `apply()`, `lapply()`, `tapply()`, `sapply()`, `vapply()` y `mapply()`.

Si no hay bucles, ¿entonces qué?

En resumen, utilice la familia de funciones `apply`; `apply()`, `lapply()`, `tapply()`, `sapply()`, `vapply()` y `mapply()`.

Observaciones

- Suele ser más fácil y/o más compacto de escribir que los bucles explícitos.
- En las primeras versiones de R también era más eficiente que los bucles, ahora comparable.

Operando con listas

- `lapply()` Siempre devuelve una lista .
- `sapply()` Intenta simplificar el retorno a un vector o matriz.
- `vapply()` Intenta simplificar a un valor de retorno preespecificado .

Section 2

Descargar y analizar información desde twitter

Paso para acceder a los datos de tweeter

1. Tener cuanta en **tweeter**
2. Instalar `rtweet` y `httpuv`
3. La autorización se realiza cuando ejecutamos algunas de las funciones:
   - *search_tweets*: Búsqueda de tweets. (En caso de tener muchos tweets se puede extraer información cada 15 minutos y limite de 18000)
   - *get_timeline*: Linea de tiempo.
   - *get_followers*: Lista de seguidores.

Descargando *tweets*

- Instalemos las siguientes librerías:

```
install.packages("rtweet")
install.packages("httpuv")
install.packages("tidyverse")
install.packages("tidytext")
install.packages("syuzhet")
install.packages("wordcloud")
```

- Llamar librerías

```
library("rtweet")
library("httpuv")
library("tidyverse")
library("tidytext")
library("syuzhet")
library("wordcloud")
```

Descargando *tweets* ¿tema del día?

```
query <- "elecciones (@petrogustavo OR @ingrodolfohdez) lang:es"
tweets <- search_tweets(q = query,
                        retryonratelimit = TRUE,
                        include_rts = FALSE,
                        n = 18000)
```

Análisis exploratorio de datos.

¿Cuál es el día de mayor actividad?

Análisis exploratorio de datos.

¿Cuál es el día de mayor actividad?

```
barplot(table(weekdays(tweets$created_at)))
```

¿Cuál es el usuario que más publica?

Análisis exploratorio de datos.

¿Cuál es el día de mayor actividad?

```
barplot(table(weekdays(tweets$created_at)))
```

¿Cuál es el usuario que más publica?

```
sort(table(tweets$screen_name),decreasing = TRUE)
```

¿tweets más popular?

Análisis exploratorio de datos.

¿Cuál es el día de mayor actividad?

```
    barplot(table(weekdays(tweets$created_at)))
```

¿Cuál es el usuario que más publica?

```
    sort(table(tweets$screen_name),decreasing = TRUE)
```

¿tweets más popular?

```
tweets[order(tweets$retweet_count,decreasing = TRUE),
       c("text", "screen_name",
         "retweet_count", "favorite_count")]

tweets[order(tweets$favorite_count,decreasing = TRUE),
       c("text", "screen_name",
         "retweet_count", "favorite_count")]
```

Analisis de sentimientos.

El **análisis de sentimientos** o la *minería de opinión* es utilizado para extraer de forma automática, información sobre la connotación negativa o positiva del lenguaje en un documento.

Diccionario de léxico NRC

El paquete `syuzhet` trabaja con cuatro diccionarios de sentimientos: *Bing*, *Afinn*, *Stanford* y *NRC*. En esta lección trabajaremos con este último puesto que es el único disponible en varios idiomas, incluido el español.

Primero paso para el *análisis de sentimientos*.

Bibliografía

- https://discdown.org/rprogramming/conditional-execution.html
- https://programminghistorian.org/es/lecciones/analisis-de-sentimientos-r
- https://rpubs.com/LucasMerolla/658511
- https://es.r4ds.hadley.nz/
- https://arcruz0.github.io/libroadp/index.html
- https://bookdown.org/gaston_becerra/curso-intro-r/