Sesión 3: Manejo de Bases de Datos y Gráficos en RStudio.

Jose Rodney Menezes De la Cruz.

rodney.menezes@pucp.edu.pe

Taller: Rstudio Aplicado a Finanzas
Colegio de Economistas de Loreto

CELOR, 2019

Índice

Comandos útiles

Comandos Útiles

- R presenta un montón de funciones para manipular las estructuras de datos:
- seq (): Genera secuencias, especificando los argumentos from, to y by.
- rep (): Replica elementos de vectores y listas.
- sort (): ordena un vector en orden ascendente. Funciona en números, pero también en cadenas de caracteres y lógicas.
- rev (): Invierte los elementos en una estructura de datos para la cual se define la inversión.
- str (): muestra la estructura de cualquier objeto.
- append (): Fusiona vectores o listas.
  - is. * (): comprueba la clase de un objeto.
  - as. * (): Convertir un objeto de una clase a otra.
- unlist (): Aplanar (posiblemente incrustado) listas para producir un vector.

Expresiones Regulares

Expresiones Regulares

- Las expresiones regulares se pueden usar para ver si existe un patrón dentro de una cadena de caracteres o un vector de cadenas de caracteres. Para ello, puede utilizar:
  - grepl (), que devuelve TRUE cuando se encuentra un patrón en la cadena de caracteres correspondiente.
  - grep (), que devuelve un vector de índices de las cadenas de caracteres que contienen el patrón.
- Ambas funciones necesitan un pattern y un argumento x, donde pattern es la expresión regular con la que se quiere hacer una comparación, y el argumento x es el vector de caracteres desde el cual se deben buscar coincidencias.

Expresiones Regulares

- Puede usar el símbolo de intercalación, ^, y el signo de dólar, $ para que coincida con el contenido ubicado en el inicio y final de una cadena, respectivamente.
- Hay otras opciones como:
- @, porque un correo electrónico válido debe contener un signo.
- . *, que coincide con cualquier carácter (.) cero o más veces (*). Puede usarlos para hacer coincidir cualquier carácter entre el signo @ y la parte ".edu" de una dirección de correo electrónico.

  \\.edu$, para que coincida con la parte ".edu" del correo electrónico al final de la cadena. La parte \\ se escapa del punto: le dice a R que desea usar el . como un caracter real.

Sentencias Condicionales

Sentencias Condicionales: if

- La estructural del condicional if es la siguiente:

```
if (condition) {
   expr
}
```

Agregar un else

- Solo puedes usar una sentencia “else” en combinación con una sentencia “if”.
- La sentencia else no requiere una condición; su código correspondiente simplemente se ejecuta si todas las condiciones anteriores en la estructura de control son FALSE.
- La estructura es la siguiente:

```
if (condition) {
   expr1
} else {
   expr2
}
```

- ¡Es importante que el comando “else” esté en la misma línea que el corchete de cierre de la parte “if”!

Else if

- La declaración “else if” le permite personalizar aún más su estructura de control. Puede agregar otras sentencias si lo desea. Tenga en cuenta que R ignora el resto de la estructura de control una vez que se ha encontrado una condición que es TRUE y se han ejecutado las expresiones correspondientes.
- La estructura es la siguiente:

```
if (condition1) {
   expr1
} else if (condition2) {
   expr2
} else if (condition3)
{ expr3
} else
{ expr4
}
```

- Es importante que lel comando “else if” estén en la misma línea que el corchete de cierre de la parte anterior de la construcción de control.

Loops

While Loop

- La estructura es la siguiente:

```
while(condition) {
    expr
}
```

- Recuerde que la parte de la condición de esta receta debe ser FALSE en algún momento durante la ejecución. De lo contrario, el bucle while continuará indefinidamente.

For Loop

- Vea la siguiente estructura:

```
primes <- c(2, 3, 5, 7, 11, 13)
```

- Versión de loop 1

```
for (p in primes) {
print(p)
}
```

- Versión de loop 2

```
for (i in 1:length(primes)) {
print(primes[i])
}
```

The Apply Family

lapply

- La estructura lapply es la siguiente:

```
lapply(X, FUN, ...)
```

- lapply toma un vector o lista X, y aplica la función FUN a cada uno de sus miembros. Si FUN requiere argumentos adicionales, páselos después de haber especificado X y FUN (...).
- El producto de lapply () es una lista, de la misma longitud que X, donde cada elemento es el resultado de aplicar FUN en el elemento correspondiente de X.

sapply

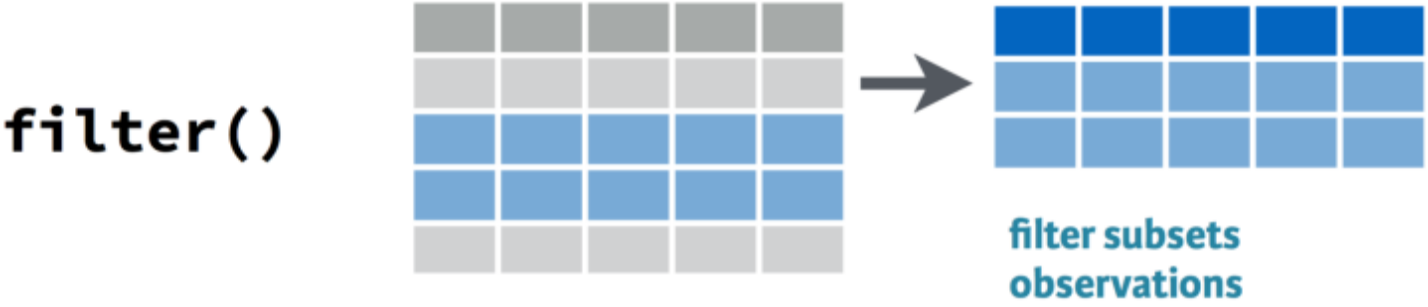
- Puede usar sapply () de manera similar a como usó lapply (). El primer argumento de sapply () es la lista o vector X sobre el que desea aplicar una función, FUN. Los posibles argumentos adicionales para esta función se especifican después (...):

```
lapply(X, FUN, ...)
```

Manipulación de Datos

Filtros

- Se debe instalar el paquete “dplyr” para usar filtros.



Mutate

- Con mutate puedes crear o agregar una nueva columna

Agrupar y Resumir

- Para obtener resultados estadísticos usamos el comando summarize