

CURSO ONLINE DE R



CURSO ONLINE

R



PhD (c)
**Ana Letycia Basso
Garcia**



PhD
**Guilherme
Kenichi Hosaka**

20
NOV

Familiarización con la interfaz de R Studio
Utilizar R Studio y el lenguaje markdown
operaciones básicas-Tipos de objetos
Exportación e importación de datos

27
Nov

Estructuras condicionales
Estructuras de repetición
Funciones de R
Elaboración de funciones

04
Dic

Instalación y aplicación de paquetes
Elaboración de gráficos sencillos
Análisis descriptivo de datos

11
Dic

Herramientas básicas para el
análisis de datos
Regresión-Análisis de varianza
Pruebas de medios



Universidade de São Paulo



ESALQ
Escola Superior de Agricultura Luiz de Queiroz
Universidade de São Paulo

CHICLAYO - 2021

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

DOSSIER DEL CURSO ONLINE DE R

SÁBADOS DE 8:00 AM -12PM

ASESORÍAS SEMANALES

CARGA HORARIA TOTAL: 32 H

Inicio: 20/11/2021 Fin: 11/12/21

Objetivos

El curso online de R será desarrollado con el propósito de quienes quieren comenzar a explorar sus datos con más autonomía y tener una aproximación con un lenguaje de programación libre.

R es el lenguaje más empleado para análisis estadísticos de datos. En este curso se propone crear una familiaridad con el ambiente de programación y que el estudiante o investigador desarrolle un razonamiento lógico en el análisis de datos. Las clases serán demostrativas (live coding) y prácticas con ejercicios para ser hechos durante la semana y entregados en la próxima clase. Se pretende que todos los participantes comprendan la sintaxis del lenguaje y sean capaces de entender y crear sus propios códigos.

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

PROFESORES

Guilherme Kenichi Hosaka



Graduado en Ciencias Biológicas por la ‘Universidad Estatal Paulista “Júlio de Mesquita Filho”’ (2011), con maestría en Fisiología y Bioquímica (2014) y con doctorado en Bioenergía (2021) por la “Escuela Superior de Agricultura “Luiz de Queiroz” (ESALQ) / Universidade de São Paulo (USP) – Brasil. Actuando en áreas de investigación como: genética, fisiología, bioquímica vegetal, análisis de genomas, transcriptomas de plantas y microrganismos, cultivo in vitro y transformación genética de plantas; laborando actualmente en el Instituto del Corazón (INCOR) del Hospital de Clínicas de la Universidad Estatal de Sao Paulo.

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Ana Letycia Basso Garcia

PhD(c) por el Programa Genética y Fitomejoramiento en el Departamento de Genética de ESALQ/USP, actuando en el Laboratorio de Bioinformática Aplicada a la Bioenergía, con estudios en análisis de datos RNA-seq, cuenta con una Maestría en Genética y Mejoramiento Vegetal (2017) egresada como Ingeniera Agrónoma por la Universidad Federal de Goiás - UFG (2014). Posee un sándwich en Iowa, Estados Unidos así mismo trabajó en Embrapa en la implementación y evaluación de pruebas multiambientales y mapeo QTL. Actualmente, trabaja en el análisis de datos NGS de plantas cultivadas para generar conocimiento y colaborar con programas en mejoramiento genético vegetal.

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

CONTENIDOS RESUMIDOS

Utilizar RStudio y el lenguaje markdown para generar informes, operaciones básicas, operaciones con vectores, tipos de objetos, estructura de datos, indexación creando un proyecto, exportación e importación de datos, funciones en R, estructuras condicionales, estructuras de repetición, familia de apply, elaboración de funciones, elaboración de gráficos sencillo, herramientas básicas para el análisis de datos, análisis de regresión, análisis de la varianza y pruebas de medios.

Programa Día 1

Conocer y familiarizarse con la interfaz de RStudio

Utilizar RStudio y el lenguaje markdown para generar informes

Hacer operaciones matemáticas básicas

Crear vectores y utilizarlos en operaciones matemáticas

Compreender y crear objetos en R

Conocer y manipular diferentes estructuras de datos, como tablas, matrices y listas

Crear y trabajar con proyectos usando R

Cargar datos en formato .txt y .csv en R para realizar análisis

Exportar tablas de datos de R para su ordenador.

Día 2

Ejecutar funciones nativas de R y crear sus propias funciones

Crear condiciones para filtrar y limpiar los datos

Elaborar y usar estructuras de repetición para automatizar procesos

Utilizar funciones de la familia apply para automatizar procesos

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Día 3

- Crear gráficos para mejor visualización e interpretación de datos
- Hacer gráficos para diagnóstico
- Crear gráficos para presentar resultados
- Personalizar gráficos con colores, líneas y formas
- Hacer paneles con múltiples gráficos para publicación

Día 4

- Realizar análisis de Regresión de datos
- Realizar análisis de varianza para conocer la significancia estadística de las variables
- Ejecutar pruebas de medios entre diversos tratamientos

CONFIDENCIALIDAD

Este material ha sido trabajado por los docentes y comité organizador que sin autorización del centro de Investigación e Innovación en Ciencias Activas Multidisciplinarias-CIICAM no puede ser plagiado ni transferido sin previa autorización.

Dirección General-CIICAM.



"INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR"

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

CENTRO DE INVESTIGACIÓN E INNOVACIÓN EN CIENCIAS ACTIVAS MULTIDISCIPLINARIAS

EQUIPO ORGANIZADOR

Ph.D. Nataly Ruiz Quiñones

M.Sc. Rene Flores Clavo

M.Sc. Gladys Angélica Apaza Castillo

M.Sc. Cristian Daniel Asmat Ortega

M.Sc. Jorge Luis Aguilar Zavaleta

MIEMBRO APOYO TÉCNICO

Bilo. Miguel Antonio Caicedo Baltodano

DOCENTES

Ph.D. Guilherme Kenichi Osaka

Ph.D. Ana Letycia Basso García

ENTIDAD COPATROCINADORA

PEZBIOTEC

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

CONTENIDO

Familiarización con la interfaz de RStudio	10
Un editor de texto	10
Una consola	11
Un panel con herramientas de rastreo	11
Un panel de archivos y previsualización	12
Utilización RStudio y el lenguaje rmarkdown para generar informes	13
Creando un R script	13
Instalación de nuevos paquetes	15
Mi primer script de R	16
Establecer un directorio de trabajo	16
Operaciones básicas	17
Operaciones con vectores	17
Objetos	19
¿Qué es un objeto?	19
Algunas operaciones con objeto	20
Buenas prácticas para nombrar objetos	22
Tipos de objetos	23
Vectores	24
Tipos de vectores	25
Interger	25
Numeric	25
Character	26
Factor	27
Logic	27
Matrices	29
Data frames (tablas)	32
Importación de datos	33
Mi primer proyecto	35
Estructuras condicionales	36
If & else (Si & sino)	36
Estructuras de repetición	39
Elaboración de gráficos sencillo	44

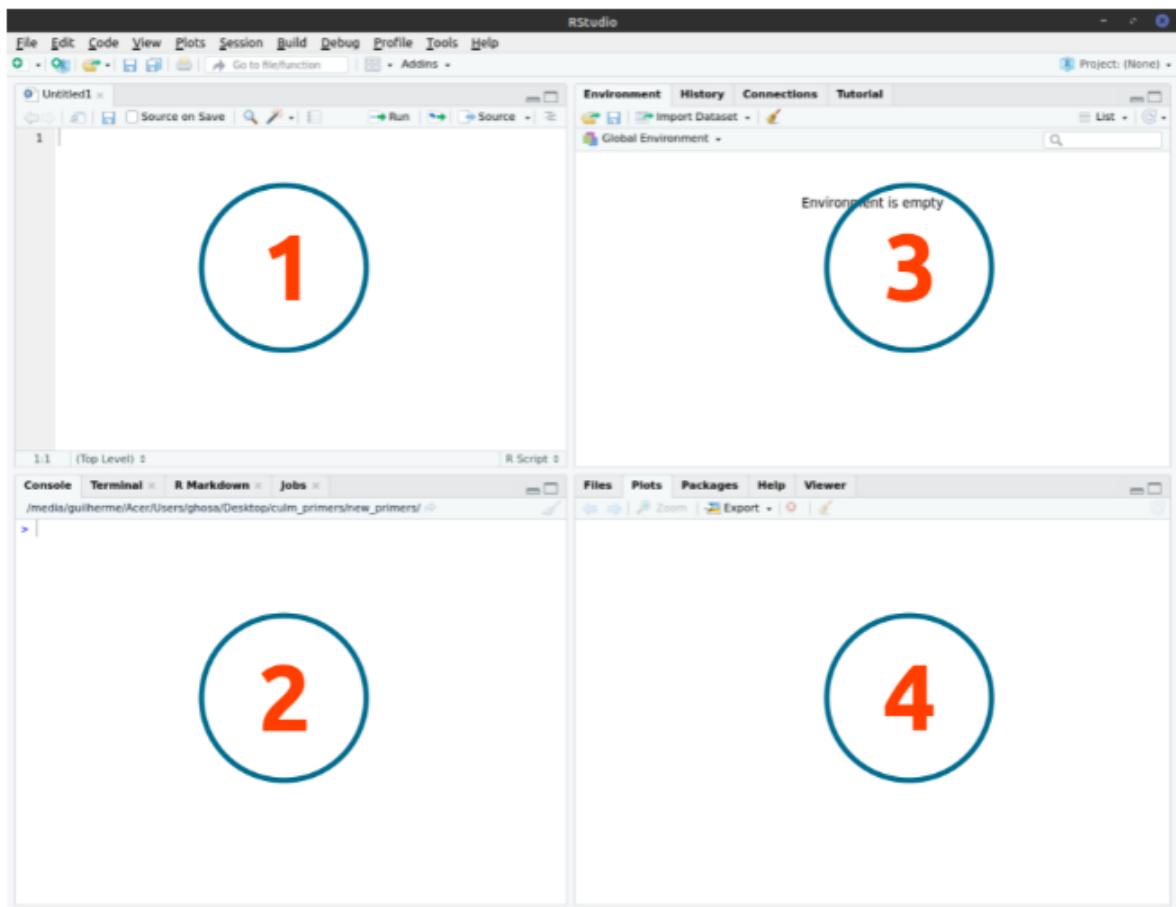
“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Gráfico de barras	44
Mapa de calor - agrupados (clusterization)	46
Gráfico de puntos	48
Guardando gráficos	49
Visualización de datos	50
Histograma	50
Gráficos de caja	50
Gráficos de violín	51
Visualización de datos 2	52
Gráfico de barras	52
Histograma	55
Diagrama de caja	56
Estadística descriptiva básica	58
Media	61
Mediana	61
Varianza	61
Máximo y Mínimo	62
Prueba de hipótesis	62
Evaluando la normalidad de los datos	63
Intervalo de confianza	64
Análisis de varianza	65
Suposiciones del análisis de varianza	65
Creando un modelo matemático	66
Diagnóstico del modelo	67
El análisis de varianza	68
Prueba de medios	69
Actividades prácticas	71

Familiarización con la interfaz de RStudio



RStudio es un ambiente de desarrollo integrado (IDE) que facilita el uso de R. También tiene varios complementos que hacen más eficiente la visualización de datos y la reproducibilidad. El programa incluye:

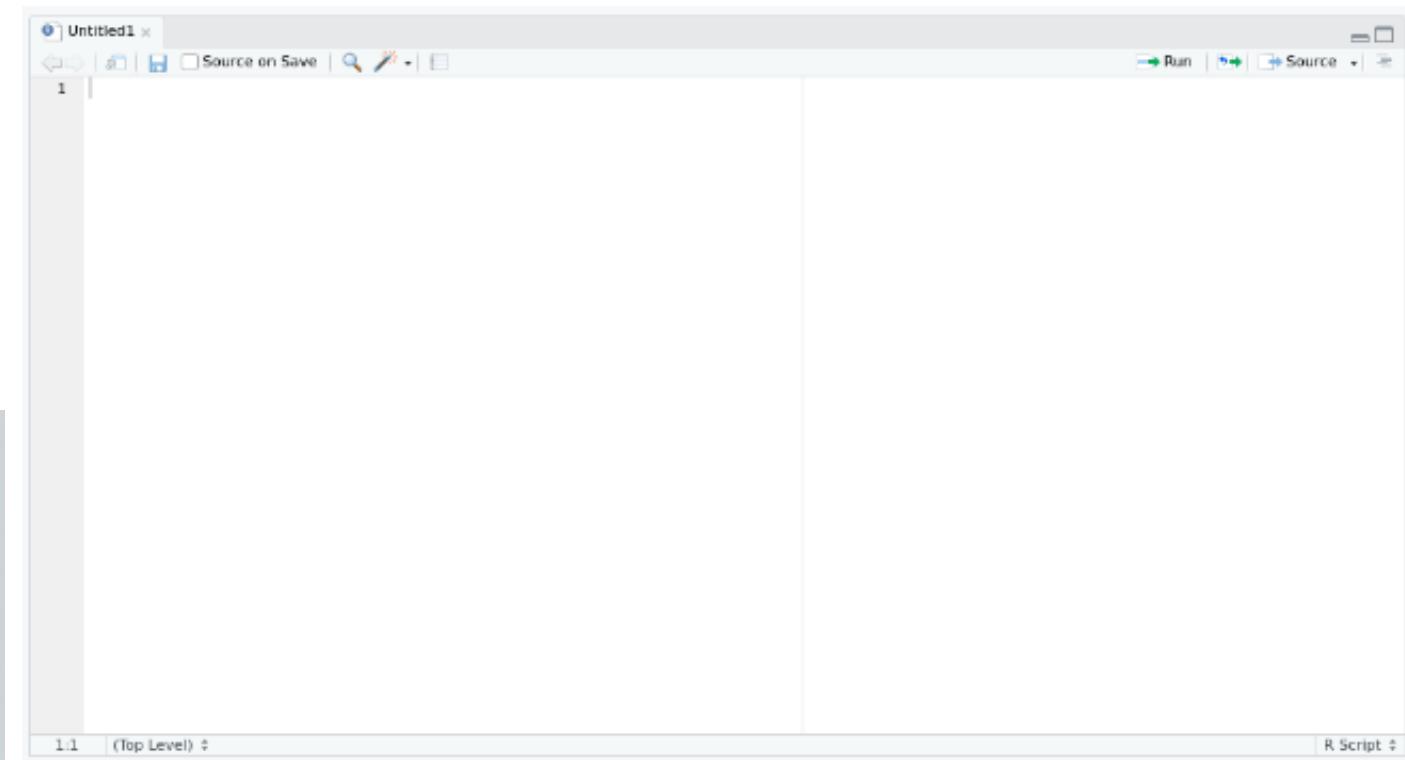
Un editor de texto

Donde escribimos nuestro código y hacemos nuestro script (conjunto de comandos a ser ejecutados). La consola de RStudio resalta la sintaxis del código y también permite la ejecución directa del mismo. Además, muestra alertas de posibles errores de sintaxis, como un paréntesis no cerrado, por ejemplo.

"INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR"

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>



Una consola

Donde se pueden realizar operaciones y ejecutar funciones directamente y también donde se muestran los resultados de los comandos ejecutados.

A screenshot of an R console window. The title bar says "Console Terminal Jobs". The R version information is displayed:
R version 4.1.0 (2021-05-18) -- "Camp Pontanezen"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

The R help text follows:
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

A single character '>' is shown at the bottom left.

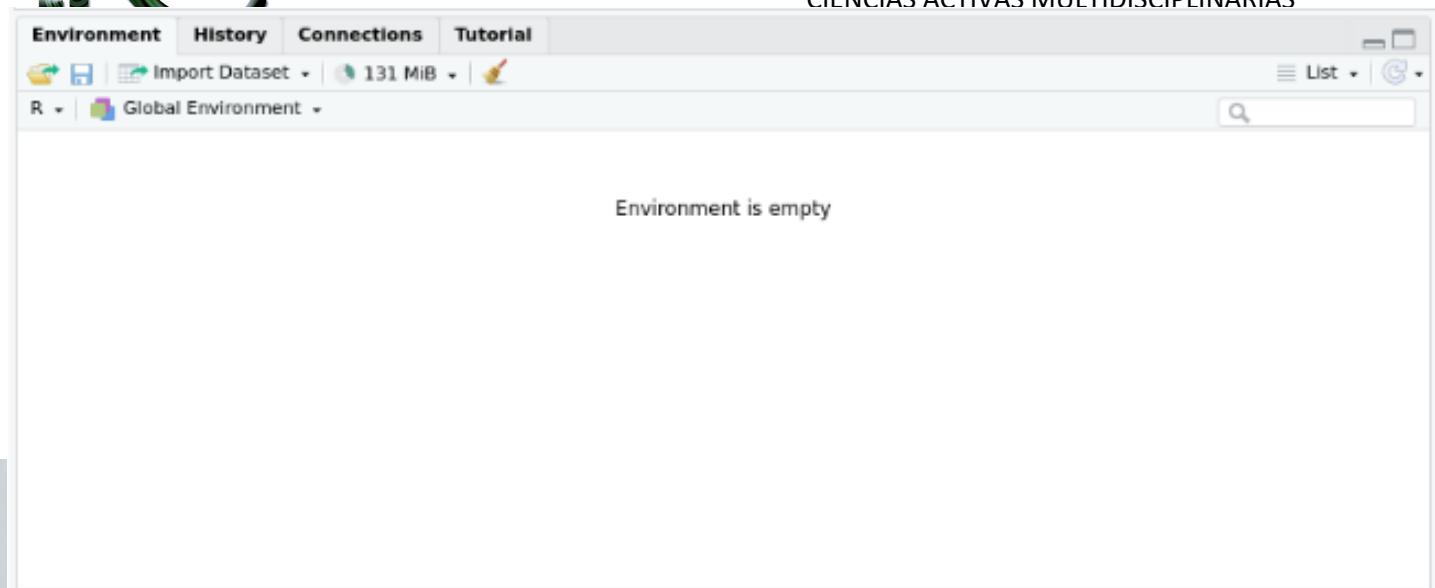
Un panel con herramientas de rastreo

Que muestra elementos del ambiente de trabajo (objetos guardados), el histórico y la depuración. Este panel permite la gestión del espacio de trabajo.

"INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR"

www.ciicam.com

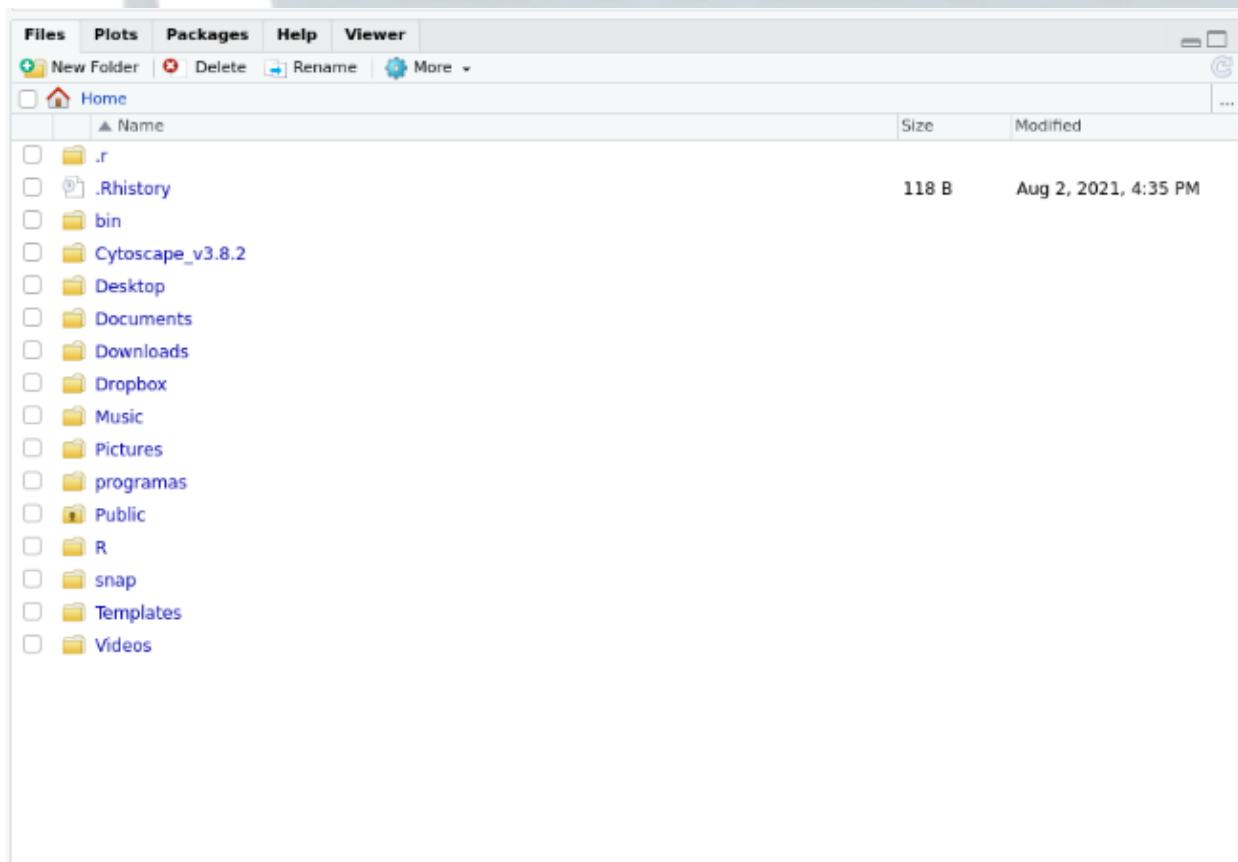
<https://ciicam.github.io/cursoR/index.html>



The screenshot shows the RStudio interface. At the top, there's a menu bar with "Environment", "History", "Connections", and "Tutorial". Below the menu is a toolbar with icons for file operations like "Import Dataset" (131 MiB), "Global Environment", and a search bar. The main workspace below the toolbar displays the message "Environment is empty".

Un panel de archivos y previsualización

Donde puede acceder fácilmente a los archivos de su directorio de trabajo, cambiar su directorio de trabajo, instalar nuevos paquetes, ver los gráficos resultantes de su código, e inclusive la previsualización de tus informes Rmarkdown (depende de la versión).



The screenshot shows the RStudio "Files" tab. The menu bar at the top includes "Files", "Plots", "Packages", "Help", and "Viewer". Below the menu is a toolbar with icons for "New Folder", "Delete", "Rename", and "More". The main area lists the contents of the current directory:

Name	Size	Modified
.r		
.Rhistory	118 B	Aug 2, 2021, 4:35 PM
bin		
Cytoscape_v3.8.2		
Desktop		
Documents		
Downloads		
Dropbox		
Music		
Pictures		
programas		
Public		
R		
snap		
Templates		
Videos		

"INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR"

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Utilización RStudio y el lenguaje rmarkdown para generar informes

En rmarkdown puede crear y procesar trozos de código de R, Python, bash (Linux) e algunas otras.

Para recibir un certificado al final del curso, tendrá que presentar informes sobre el seguimiento de las actividades prácticas

Recomendamos crear un Rscript y probar el código y luego reemplazarlo en rmarkdown.

Creando un R script



Para crear un R script haga clic



en el menú principal y seleccione

Ahora puede probar los fragmentos de código para reemplazar en rmarkdown y generar sus informes.



Para crear un Rmarkdown archivo haga clic



en el menú principal y seleccione

En la ventana abierta puede introducir la información del Título ("Día 1", por ejemplo) y del Nombre. Podemos generar informes en HTML, PDF o Word. En este curso sólo cubriremos la salida HTML.

Al final tendrá algo similar a esta figura:

"INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR"

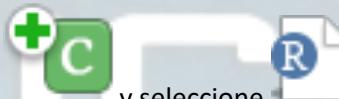
www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>



```

1: ...
2: title: "Gila s"
3: author: "Guthérne"
4: date: "8/7/2021"
5: output: html_document
6: ...
7:
8: ##(n, setup, include=FALSE)
9: knitr::opts_chunk$echo = TRUE
10: ...
11:
12: # R Markdown
13:
14: This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
15:
16: When you click the Knit button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:
17:
18: ##(n, cars)
19: summary(cars)
20: ...
21:
22: # Including Plots
23:
24: You can also embed plots, for example:
25:
26: ##(r pressure, echo=FALSE)
27: plot(pressure)
28: ...
29:
30: Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.
31:
32: [Din] +
```



Para crear un *chunk* (trozo de código) haga clic en  y seleccione 

Su ventana se verá así, donde puede insertar el código probado en el R script. También hemos insertado ejemplos de código que pueden verse como texto en el archivo html. Siéntase libre de editar esta parte del texto.

Si quieres probar que el código funciona perfectamente, puedes hacer clic en el botón  situado en la esquina superior derecha del *chunk*.



```

1: ...
2: title: "Untitled"
3: author: "Guthérne"
4: date: "8/7/2021"
5: output: html_document
6: ...
7:
8: # Subtítulo principal
9:
10: Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
11:
12: # Subtítulo secundario
13:
14: * Lísta:
15:   * Un
16:   * Dos
17:   * Tres
18: * Más
19:
20: # Negritas
21:
22: # Índice
23:
24: ##(r)
25:
26: # INSERTE SU R SCRIPT AQUÍ
27:
28: ...
29:
30:
```

Si no aparece ningún error, puede compilar el archivo haciendo clic en el botón  situado en la barra principal. Elija su ubicación preferida para guardar el archivo HTML generado.

Para más detalles sobre el formato rmarkdown puede consultar el tutorial de [Rpubs \(En español\)](#).

Instalación de nuevos paquetes

Si no ha instalado un paquete necesario para un proceso. Puede instalar los paquetes utilizando el comando en la consola o en el editor de texto. Preferiblemente en la consola porque la instalación se hace una sola vez.

Si quieras generar tus informes en formato PDF, necesitarás instalar látex. Para más detalles, [pulse aquí](#).

```
install.packages("tinytex", dependencies = TRUE)
```

Se recomienda ejecutar la instalación del paquete con la opción de dependencias TRUE para instalar las dependencias del paquete automáticamente.

Y para cargar el paquete instalado, ejecute el siguiente comando al inicio de su script:

```
library(tinytex)
```

Sin comillas. En este caso, rmarkdown identificará automáticamente el paquete, pero en otros casos es necesario cargarlo en el inicio del script.

A veces el sistema operativo Windows requiere la instalación de Rtools para algunos paquetes. Si necesitas instalarlo puedes encontrarlo [aquí](#).

Mi primer script de R

Establecer un directorio de trabajo

Para facilitar el trabajo con los scripts de R, establecemos el directorio de trabajo al principio del script.

Para Windows: `setwd("C:/Users/MyName/Desktop/")`

Para Linux: `setwd("~/Desktop/")`

Para Mac: `setwd("~/Desktop/")`

Una tecla muy utilizada en R es el **TAB**, con el que R le ayudará a completar o sugerir algunas opciones de funciones o parámetros.

En el caso de la función setwd, TAB puede ayudarle a llenar la ruta del libro de trabajo.

Comentarios

Utilice el "#" para comentar el código. Las partes presentes después de "#" no se ejecutarán.

Operaciones básicas

R como cualquier otro lenguaje de programación puede funcionar como una calculadora. Las operaciones básicas son muy similares a las de Excel, por ejemplo.

```
2+3 #Sumar
2*3 #Multiplicar
2/3 #Dividir
2^3 #Elevación al cubo* de un número
3^(1/2) #Raíz cuadrada*
```

Funciones de R

```
sqrt(9) #Raíz cuadrada
log(100, base = 10) #Logaritmo en base 10
log(100) #Logaritmo con base neperiana
```

¡Manos a la obra!

Practicamos lo que acabamos de aprender. Utilice paréntesis () para priorizar las operaciones:

$$\left(\frac{15 + 6.4 - 5}{7} \right) + \sqrt{256}$$

Resultado:

```
## [1] 18.34286
```

Operaciones con vectores

Las mismas operaciones básicas que se muestran en la sección anterior se pueden realizar como un vector. Para crear un vector, usamos la función `c` (*Combine values*).

```
c(1,2,3,4,5,6,7,8,9,10)
## [1] 1 2 3 4 5 6 7 8 9 10
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

De la misma manera podemos usar : para números secuenciales.

```
1:10
```

```
## [1] 1 2 3 4 5 6 7 8 9 10
```

Otra función para generar secuencias es la **seq**.

```
seq(from=0, to=50, by=2)  
seq(0,50,2) #Respetando la orden de los parámetros
```

```
## [1] 0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50
```

¡Manos a la obra!

Practicamos lo que acabamos de aprender. Cree una secuencia usando la función seq que va de 15 a 50, con intervalos de 7 a 7.

Resultado:

```
## [1] 15 22 29 36 43 50
```

Números nulos o secuencias perdidas (NA)

Utilice el parámetro **na.rm = T** de la función para no tener en cuenta los valores perdidos.

La función **rep** genera secuencias con números repetidos:

```
rep(1:4, 3)
```

```
## [1] 1 2 3 4 1 2 3 4 1 2 3 4
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Ahora que hemos aprendido algunas funciones útiles, realicemos operaciones vectoriales:

```
c(1,2,3,4,5)*2
```

```
## [1] 2 4 6 8 10
```

```
c(1,2,3,4,5)+c(1,2,3,4,5)
```

```
## [1] 2 4 6 8 10
```

```
c(1,2,3,4,5)*c(1,2,3,4,5)
```

```
## [1] 1 4 9 16 25
```

Objetos

Para hacer menos repeticiones y ser más práctico, vamos a asignar los resultados de nuestras operaciones a variables, que en R las llamamos objetos.

¿Qué es un objeto?

Cualquier variable o conjunto de variables que asignemos a un espacio concreto de la memoria del ordenador mientras estamos utilizando R será un objeto, o un conjunto de objetos. Es decir, estos objetos aparecerán en el ambiente R dentro de Data.

```
x = 1:4
```

Para acceder valores en un objeto:

```
x
```

```
## [1] 1 2 3 4
```

Algunas operaciones con objeto

Multiplicando el objeto por un número:

```
x * 2
```

```
## [1] 2 4 6 8
```

Puedes hacer cualquier operación con su objeto, incluido utilizar-lo en funciones del R.

```
rep(x, 2)
```

```
## [1] 1 2 3 4 1 2 3 4
```

Y también puedes salvar los resultados de las operaciones con objetos en otros objetos.

```
y = x * 2
```

Para hacer operaciones con dos objetos, el R los alinea y realiza la operación elemento a elemento.

```
x + y
```

```
## [1] 3 6 9 12
```

Si los objetos son de distinto tamaño, se repetirán todos los elementos del objeto más pequeño para realizar la operación elemento a elemento con todos los elementos del objeto más grande.

```
x*2
```

```
## [1] 2 4 6 8
```

```
x*c(2,2)
```

```
## [1] 2 6 6 12
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Y si el vector más pequeño no es un múltiplo del vector más grande, se recibe un aviso.

```
x*c(2,3,4)
```

```
## Warning in x * c(2, 3, 4):
```

```
la longitud del objeto más grande no es múltiplo de la longitud del objeto más pequeño
```

```
## [1] 2 6 12 8
```

Puedes aplicar funciones en el objeto:

```
sum(x)
```

```
## [1] 10
```

```
mean(x)
```

```
## [1] 2.5
```

```
var(x)
```

```
## [1] 1.666667
```

Y almacenar los resultados en un otro objeto:

```
z = sum(x)
```

```
x = x + x
```

```
## [1] 2 4 6 8
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

El lenguaje es sensible a letras mayúsculas y minúsculas:

```
x
```

Buenas prácticas para nombrar objetos

- Pueden contener letras, números, “.” y “_”, pero no pueden ser solo números y no pueden empezar con número o punto
- No pueden contener espacios
- No utilice acentos
- Evite nombres de funciones del R, como sum, var, df
- Atención para letras mayúsculas y minúsculas:

`mis_datos ≠ Mis_Datos ≠ MIS_DATOS`

- No use nombres que no remeten a que se está analizando:

```
a = mean(x)
```

¿Qué es a? En dos meses no te acordarás.

```
mean_x = mean(x)
```

Para saber qué objetos se guardan en la memoria, puedes utilizar la función ls().

```
ls()
```

```
## [1] "a"    "mean_x" "x"    "y"    "z"
```

Para remover un objeto:

```
rm(x3)
```

```
## Warning in rm(x3): objeto 'x3' não encontrado
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Para remover todos los objetos:

```
#rm(list = ls())
```

¡Atención! El código arriba removerá todos los objetos en el workspace de R, por eso no lo ejecuté.

Los objetos se guardan provisionalmente en la RAM mientras se utiliza el R. Para guardarlos efectivamente, es necesario almacenarlos en un archivo en formato Rdata. Para esto se emplea la función *save()*.

```
save(file = "mi_objeto.RData", x)
```

Para salvar todos los objetos del ambiente de trabajo:

```
save.image(file = "mi_ambiente.RData")
```

Tipos de objetos

¡En R todo es un objeto! Eses objetos pueden ser de distintas formas. Es muy común lidiar con distintos tipos de objeto cuando se está haciendo una analice en R y por eso, es imperativo conocer cada uno de los tipos y como hacer las conversiones necesarias. Cuando decimos tipos de objetos nos referimos a los tipos de datos que componen los vectores que forman un objeto y a la estructura del objeto. [VER AQUÍ](#)

Eso es importante porque determina como el objeto será manipulado por las funciones.

Los datos pueden ser estructurados de distintas maneras:

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Vea que las estructuras de datos son vectores o combinaciones de vectores.

Vectores

Un vector puede contener apenas un tipo de datos, pero varios vectores de distintos tipos, pueden ser organizados en distintas estructuras de datos, como tablas, matrices, listas y otros.

```
v = c("ABCD", 2:5, TRUE)
```

```
class(v)
```

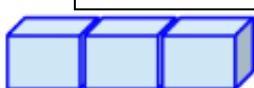
```
## [1] "character"
```

```
str(v)
```

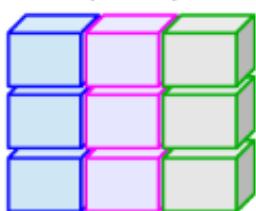
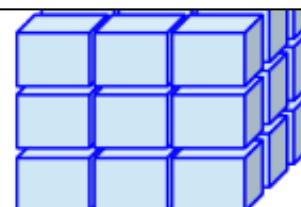
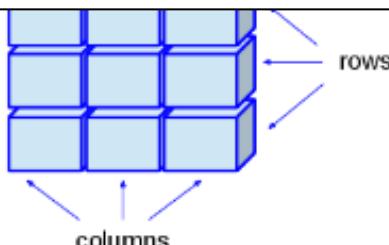
Matrix

Array

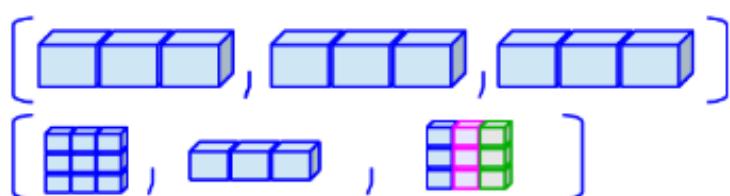
```
## chr [1:6] "ABCD" "2" "3" "4" "5" "TRUE"
```



Data Frame
(Table)



Lists



Tipos de vectores

Integer

```
i = seq(1:10)
```

```
i
```

```
## [1] 1 2 3 4 5 6 7 8 9 10
```

```
class(i)
```

```
## [1] "integer"
```

```
is.integer(i)
```

```
## [1] TRUE
```

Numeric

```
(n = rep(c(1.1, 2.3, 4.8, 7.0, 10.0), 3))
```

```
## [1] 1.1 2.3 4.8 7.0 10.0 1.1 2.3 4.8 7.0 10.0 1.1 2.3 4.8 7.0 10.0
```

```
class(n)
```

```
## [1] "numeric"
```

```
str(n)
```

```
## num [1:15] 1.1 2.3 4.8 7 10 1.1 2.3 4.8 7 10 ...
```

```
is.integer(n)
```

```
## [1] FALSE
```

```
is.numeric(n)
```

```
## [1] TRUE
```

Podemos transformar un objeto en una clase distinta

```
i2 = as.numeric(i)
```

```
class(i2)
```

```
## [1] "numeric"
```

```
is.integer(i2)
```

```
## [1] FALSE
```

```
is.numeric(i2)
```

```
## [1] TRUE
```

Los objetos de tipo *integer* ocupan menos espacio en la memoria que los objetos de tipo *numeric*

```
object.size(i) # integer
```

```
## 96 bytes
```

```
object.size(i2) # numeric
```

```
## 176 bytes
```

Character

```
nombres = c("Ana", "Maria", "José", "Juan", "Carlos", "Julia")
```

```
str(nombres)
```

```
## chr [1:6] "Ana" "Maria" "José" "Juan" "Carlos" "Julia"
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Factor

Un factor es un vector predefinido que almacena datos categóricos.

```
formacion = c("Biología", "Ingeniería", "Sociología", "Ingeniería", "Biología", "Geología")
```

```
str(formacion)
```

```
## chr [1:6] "Biología" "Ingeniería" "Sociología" "Ingeniería" "Biología" ...
```

```
formacion_factor = as.factor(formacion)
```

```
str(formacion_factor)
```

```
## Factor w/ 4 levels "Biología", "Geología", ... : 1 3 4 3 1 2
```

```
levels(formacion_factor)
```

```
## [1] "Biología" "Geología" "Ingeniería" "Sociología"
```

```
length(formacion_factor)
```

```
## [1] 6
```

Logic

Retomando los vectores i y i2 para algunos comparativos

```
str(i)
```

```
## int [1:10] 1 2 3 4 5 6 7 8 9 10
```

```
str(i2)
```

```
## num [1:10] 1 2 3 4 5 6 7 8 9 10
```

¿El Vector i es igual al vector i2?

```
i == i2
```

```
## [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

"INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR"

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

¿El primer elemento del vector i es igual al cuarto elemento del vector i?

```
i[1] == i[4]
```

```
## [1] FALSE
```

¿Es más grande?

```
i[1] > i[4]
```

```
## [1] FALSE
```

¿Es más pequeño?

```
i[1] < i[4]
```

```
## [1] TRUE
```

¿El vector i es diferente de dos?

```
i != 2
```

```
## [1] TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

¿El vector i es igual o más grande que dos?

```
i >= 2
```

```
## [1] FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

¿El vector i es más grande que dos y más pequeño que cinco?

```
## [1] FALSE FALSE TRUE TRUE FALSE FALSE FALSE FALSE FALSE
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

¿El vector i es más pequeño que dos o más pequeño que cinco?

```
i < 2 | i < 5
```

```
## [1] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE
```

¿Cuál es la posición de los elementos del vector i que son más grandes que 8?

```
which(i > 8)
```

```
## [1] 9 10
```

Podemos buscar elementos específicos en un vector:

```
formacion %in% c("Biología")
```

```
## [1] TRUE FALSE FALSE FALSE TRUE FALSE
```

Y podemos extraer la información que está en una posición que estamos interesados. Eso se llama indexación y la estudiaremos más adelante también.

¡Manos a la obra!

Cree un vector con los nombres de cinco personas que admirás. ¿Qué tipo de datos forman este vector? ¿Qué nombre se almacena en la tercera posición?

Matrices

Las matrices se componen de líneas y columnas, generalmente formadas por elementos numéricos. Permiten realizar grandes operaciones de forma automatizada.

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Para crear una matriz:

```
X = matrix(1:10, nrow = 5, ncol = 2)
```

```
X
```

```
## [,1] [,2]  
## [1,] 1 6  
## [2,] 2 7  
## [3,] 3 8  
## [4,] 4 9  
## [5,] 5 10
```

Vea que los números se distribuyen primer en las columnas por defecto. Si quieres cambiar eso:

```
W = matrix(1:10, nrow = 5, ncol = 2, byrow = TRUE)
```

```
W
```

```
XW = matrix(c(X, W), nrow = 5, ncol = 4)
```

```
## [,1] [,2]  
## [1,] 1 2  
## [2,] 3 4  
## [3,] 5 6  
## [4,] 7 8  
## [5,] 9 10
```

Se puede crear matrices también con objetos ya almacenados en la memoria (vectores o hasta mismo otras matrices):

Podemos hacer operaciones matriciales, como la multiplicación por un número.

```
X*2
```

```
## [,1] [,2]
## [1,] 2 12
## [2,] 4 14
## [3,] 6 16
## [4,] 8 18
## [5,] 10 20
```

O la multiplicación con otra matriz elemento a elemento.

```
X*X
```

```
## [,1] [,2]
## [1,] 1 36
## [2,] 4 49
## [3,] 9 64
## [4,] 16 81
## [5,] 25 100
```

Y también operaciones matriciales.

```
X%*%t(X)
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

```
## [1] [,1] [,2] [,3] [,4] [,5]
## [1,] 37 44 51 58 65
## [2,] 44 53 62 71 80
## [3,] 51 62 73 84 95
## [4,] 58 71 84 97 110
## [5,] 65 80 95 110 125
```

Realizar operaciones con matrices exige conocimientos previos de álgebra matricial. Puede explorar la sintaxis de R para este tipo de operaciones. Para aprender más sobre el álgebra matricial, puedes ver [este material](#).

Data frames (tablas)

Las tablas nos permiten almacenar diferentes tipos de datos en sus columnas. Estas tablas de datos de R son similares a las tablas creadas en otros programas, como las hojas de cálculo de Excel.

Para crear una tabla directamente en R:

```
paises <- data.frame("Pais" = c("Peru", "Argentina", "Chile", "Mexico", "Estados Unidos",
"Inglaterra", "Australia"),
"Continente" = c("América Latina", "América Latina", "América Latina",
"América Central", "América del Norte", "Europa", "Oceania"))
paises
```

	País	Continente
## 1	Peru	América Latina
## 2	Argentina	América Latina
## 3	Chile	América Latina
## 4	Mexico	América Central
## 5	Estados Unidos	América del Norte
## 6	Inglaterra	Europa
## 7	Australia	Oceania

Podemos acceder a cada una de las columnas con:

```
paises$Continente
```

```
## [1] "América Latina"  "América Latina"  "América Latina"  
## [4] "América Central" "América del Norte" "Europa"  
## [7] "Oceania"
```

O también pelo número de la columna:

```
paises[,2]
```

```
## [1] "América Latina"  "América Latina"  "América Latina"  
## [4] "América Central" "América del Norte" "Europa"  
## [7] "Oceania"
```

Cuando utilizamos corchetes, podemos definir las posiciones de los elementos a que queremos acceder dentro de la tabla. El primer valor dentro del corchete se refiere a la línea y el segundo valor, a la columna.

```
paises[1,]
```

```
## País  Continente  
## 1 Peru América Latina
```

```
paises[2,2]
```

```
## [1] "América Latina"
```

Importación de datos

"INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR"

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Para realizar un análisis de datos en el mundo real, probablemente necesitará un conjunto de datos recogidos previamente que desee evaluar. Hay muchas formas de importar tablas de datos en R y aquí hablaremos de las más comunes.

read.table

Puedes ingresar archivos tipo texto

```
mi_tabla = read.table("tabelas/Ant-Man.And.The.Wasp.txt", sep = "\t")
```

read.csv

Para leer archivos separados por "," (.csv)

```
mi_data = read.csv("tabelas/felicidad_américa_latina.csv")
str(mi_data)
```

```
## 'data.frame': 20 obs. of 5 variables:
## $ Country.name : chr "Costa Rica" "Guatemala" "Uruguay" "Brazil" ...
## $ Regional.indicator : chr "Latin America and Caribbean" "Latin America and Caribbean" "Latin America and Caribbean" "Latin America and Caribbean" ...
## $ Ladder.score : num 7.07 6.43 6.43 6.33 6.32 ...
## $ Social.support : num 0.891 0.813 0.925 0.882 0.831 0.877 0.896 0.882 0.762 0.847 ...
## $ Healthy.life.expectancy: num 71.4 65 69.1 66.6 68.6 ...
```

Es muy importante observar los argumentos de la función read.csv y también de read.table().

```
#?read.csv()
```

```
mi_data = read.csv(file = "tabelas/felicidad_américa_latina.csv", header = T, sep = ",", dec = ".")
```

```
str(mi_data)
```

```
## 'data.frame': 20 obs. of 5 variables:  
## $ Country.name : chr "Costa Rica" "Guatemala" "Uruguay" "Brazil" ...  
## $ Regional.indicator : chr "Latin America and Caribbean" "Latin America and Caribbean" "Latin America and Caribbean" "Latin America and Caribbean" ...  
## $ Ladder.score : num 7.07 6.43 6.43 6.33 6.32 ...  
## $ Social.support : num 0.891 0.813 0.925 0.882 0.831 0.877 0.896 0.882 0.762  
0.847 ...  
## $ Healthy.life.expectancy: num 71.4 65 69.1 66.6 68.6 ...
```

¡Atención!

Se utiliza el Excel, puedes pasar de su archivo csv sea separado por ";" o sus casas decimales son separadas por ",". Se define adecuadamente los argumentos da función, podrá importar esos archivos sin problemas. Así, se ve que el resultado de las funciones head() o str() parece inadecuado, verifica cuáles son los separadores de su archivo e intenta nuevamente la importación definiendo los argumentos sep y dec de la función read.csv().

Mi primer proyecto

Con RStudio puedes mantener tu trabajo organizado por proyectos. Así, si estás haciendo un análisis de datos para tu tesis, por ejemplo, puedes crear un proyecto para mantener todos los datos, el código, los resultados, el historial de análisis y los informes en un único directorio de fácil acceso.

Podemos hacer de este curso un proyecto en R. Así, todo lo que aprendas en estos días será fácil de encontrar y estudiar cuando quieras.

Te guiaré para crear tu primer proyecto en R.

Estructuras condicionales

If & else (Si & sino)

Estructuras condicionales son dispositivos de lenguaje de programación para determinar qué bloque de código se ejecutará en una condición determinada.

```
promedio = 12

if (promedio >= 11) {
  print("Passaste de año!")
} else{
  print("Repetiste de año!")
}
```

En ejemplo de arriba, verificamos si su objeto promedio es mayor que 11 con la condición *if*, de lo contrario, el *else* sería activado.

```
promedio = 8

if (promedio >= 11) {
  print("Passaste de año!")
} else{
  print("Repetiste de año!")
}
```

```
## [1] "Repetiste de año!"
```

Cuando necesite usar más de una condición, puede usar *else if*.

```

promedio = 8

if (promedio >= 11) {

  print("Passaste de año!")

} else if (promedio < 11 & promedio > 6) {

  print("Necessitas recuperación!")

} else{

  print("Repetiste de año!")

}

```

```
## [1] "Necessitas recuperación!"
```

En esta tabla podemos ver la encuesta del índice de felicidad, apoyo social y esperanza de vida de los países latinoamericanos en el año 2021. Datos de felicidad descargados de [kaggle](#).

Country.name	Regional.indicator	Ladder.score	Social.support	Healthy.life.expectancy
Costa Rica	Latin America and Caribbean	7.069	0.891	71.4
Guatemala	Latin America and Caribbean	6.435	0.813	64.958
Uruguay	Latin America and Caribbean	6.431	0.925	69.1
Brazil	Latin America and Caribbean	6.33	0.882	66.601
Mexico	Latin America and Caribbean	6.317	0.831	68.597
Jamaica	Latin America and Caribbean	6.309	0.877	67.5
Panama	Latin America and Caribbean	6.18	0.896	69.652

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Country.name	Regional.indicator	Ladder.score	Social.support	Healthy.life.expectancy
Chile	Latin America and Caribbean	6.172	0.882	70
El Salvador	Latin America and Caribbean	6.061	0.762	66.402
Colombia	Latin America and Caribbean	6.012	0.847	68.001
Nicaragua	Latin America and Caribbean	5.972	0.864	67.657
Argentina	Latin America and Caribbean	5.929	0.898	69
Honduras	Latin America and Caribbean	5.919	0.812	67.3
Peru	Latin America and Caribbean	5.84	0.832	68.25
Ecuador	Latin America and Caribbean	5.764	0.821	68.8
Bolivia	Latin America and Caribbean	5.716	0.81	63.901
Paraguay	Latin America and Caribbean	5.653	0.893	65.9
Dominican Republic	Latin America and Caribbean	5.545	0.853	66.102
Venezuela	Latin America and Caribbean	4.892	0.861	66.7
Haiti	Latin America and Caribbean	3.615	0.54	55.7

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

¡Manos a la obra!

El promedio del índice de la felicidad (columna ladder.score) de los países latinoamericanos es de 5.90.

Cree una condición para identificar los países que tienen un índice superior a 5.90 y print "País feliz!", sino print "Más fiestas!". Para este ejercicio, descargue la tabla [aquí](#).

Resultado esperado:

¿Qué sucedió? ¿Cuál es la diferencia entre el promedio de notas y este caso?

Estructuras de repetición

Si tiene que repetir una secuencia de instrucciones varias veces hasta que alcance una determinada condición, puede utilizar la función *for*.

```
for(i in 1:10){  
  print(i)  
}
```

```
## [1] 1  
## [1] 2  
## [1] 3  
## [1] 4  
## [1] 5  
## [1] 6  
## [1] 7  
## [1] 8  
## [1] 9  
## [1] 10
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Ahora podemos comprobar los índices de la tabla de felicidad.

```
for (i in 1:nrow(tabla)){
  if (tabla$Ladder.score[i] > 5.90){
    print("País feliz!")
  } else {
    print("Más fiestas!")
  }
}
```



“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

```
## [1] "País feliz!"  

## [1] "Más fiestas!"  

## [1] "Más fiestas!"
```

También podemos crear una nueva columna para ingresar nuevos datos

```
for (i in 1:nrow(tabela)){  

  if (tabela$Ladder.score[i] > 5.90){  

    tabela$felicidad[i] = "País feliz!"  

  } else {  

    tabela$felicidad[i] = "Más fiestas!"  

  }
}
```

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

##	Country.name	Regional.indicator	Ladder.score	Social.support
## 1	Costa Rica Latin	America and Caribbean	7.069	0.891
## 2	Guatemala	Latin America and Caribbean	6.435	0.813
## 3	Uruguay	Latin America and Caribbean	6.431	0.925
## 4	Brazil	Latin America and Caribbean	6.330	0.882
## 5	Mexico	Latin America and Caribbean	6.317	0.831
## 6	Jamaica	Latin America and Caribbean	6.309	0.877
## 7	Panama	Latin America and Caribbean	6.180	0.896
## 8	Chile	Latin America and Caribbean	6.172	0.882
## 9	El Salvador	Latin America and Caribbean	6.061	0.762
## 10	Colombia	Latin America and Caribbean	6.012	0.847
## 11	Nicaragua	Latin America and Caribbean	5.972	0.864
## 12	Argentina	Latin America and Caribbean	5.929	0.898
## 13	Honduras	Latin America and Caribbean	5.919	0.812
## 14	Peru	Latin America and Caribbean	5.840	0.832
## 15	Ecuador	Latin America and Caribbean	5.764	0.821
## 16	Bolivia	Latin America and Caribbean	5.716	0.810
## 17	Paraguay	Latin America and Caribbean	5.653	0.893
## 18	Dominican Republic	Latin America and Caribbean	5.545	0.853
## 19	Venezuela	Latin America and Caribbean	4.892	0.861
## 20	Haití	Latin America and Caribbean	3.615	0.540
##	Healthy.life.expectancy felicidad			
## 1	71.400 País feliz!			
## 2	64.958 País feliz!			
## 3	69.100 País feliz!			
## 4	66.601 País feliz!			
## 5	68.597 País feliz!			
## 6	67.500 País feliz!			
## 7	69.652 País feliz!			
## 8	70.000 País feliz!			
## 9	66.402 País feliz!			
## 10	68.001 País feliz!			
## 11	67.657 País feliz!			
## 12	69.000 País feliz!			
## 13	67.300 País feliz!			
## 14	68.250 Más fiestas!			
## 15	68.800 Más fiestas!			
## 16	63.901 Más fiestas!			
## 17	65.900 Más fiestas!			
## 18	66.102 Más fiestas!			
## 19	66.700 Más fiestas!			
## 20	55.700 Más fiestas!			

También hay una función llamada **while**, que es muy similar a *for*.

```
x = 0
while(x < 10){
  x = x + 1
  print(x)
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
```

Elaboración de gráficos sencillo

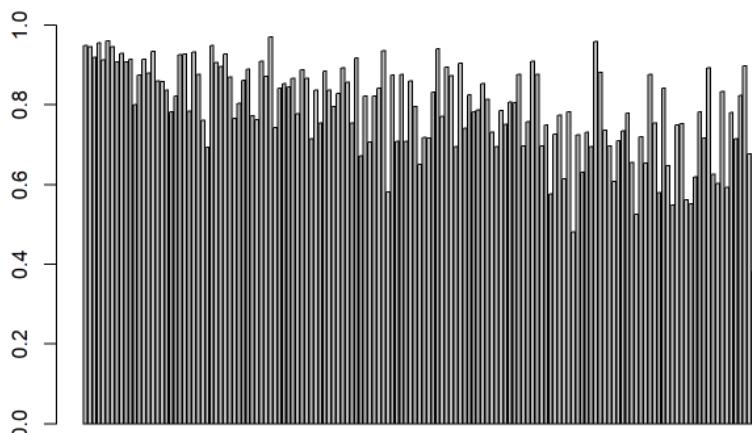
Vamos a generar algunos gráficos sencillos utilizando las funciones básicas de R. Hay paquetes como [ggplot2](#), [plotly](#) y shiny que tienen herramientas mucho más detalladas para la construcción de gráficos, pero requieren un poco más de tiempo para aprender su sintaxis. En este paso utilizaremos datos de algunos países del mundo. Para obtenerlos [haga clic aquí](#).

```
happines =  
read.csv("C:/Users/ghosa/OneDrive/Documentos/GitHub/cursoR/tabelas/world-  
happiness-report-2021.csv", sep = ",")
```

Gráfico de barras

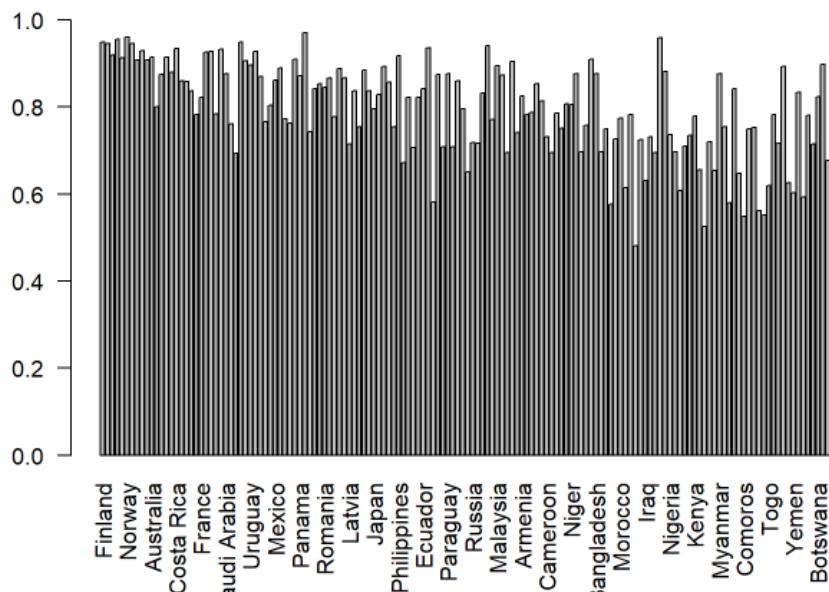
Un gráfico de barras puede servir para mostrar la asociación entre una variable numérica y una variable categórica. Por ejemplo, podemos visualizar el índice de libertad de decisión (variable numérica) por país (variable categórica).

```
barplot(happines$Freedom.to.make.life.choices, ylim=c(0,1))
```



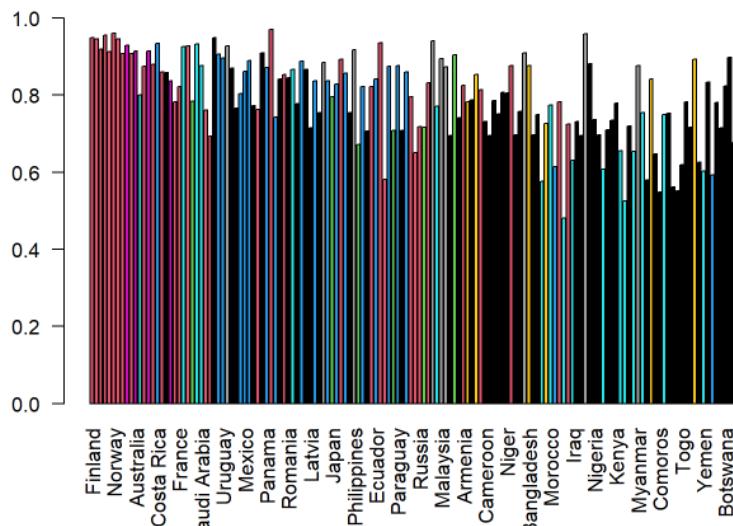
Arriba podemos ver un gráfico sencillo, vamos a intentar mejorarlo un poco poniendo los nombres de los países. En este caso utilizaremos los parámetros **names.arg** para los nombres de los países y las para girar 90 grados.

```
barplot(happines$Freedom.to.make.life.choices, ylim=c(0,1),
        names.arg=happines$i..Country.name, las=2)
```



Ahora vamos a intentar colorear las barras de diferentes colores según los países. Para ello utilizamos el parámetro **col** (acepta valores numéricos para los colores o factores).

```
barplot(happines$Freedom.to.make.life.choices, ylim=c(0,1),
        names.arg=happines$i..Country.name, las=2, col = as.factor(happines$Regional.indicator))
```



¡Manos a la obra!

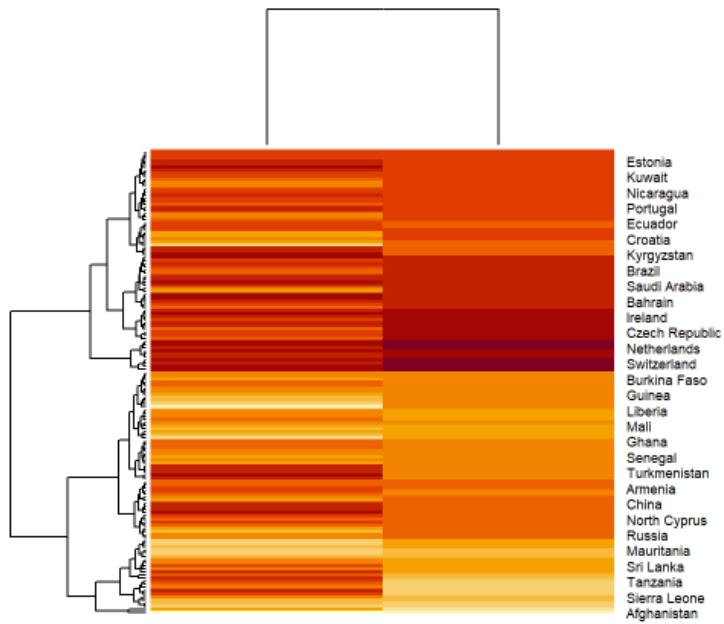
Cree un gráfico de barras con los percepción de corrupción (*Perceptions of corruption*) por Indicador de Región (*Regional Indicator*).

Mapa de calor - agrupados (clusterization)

Un mapa de calor (o heatmap) es una representación gráfica de datos en la que los valores se representan por colores. Los mapas de calor facilitan la visualización de datos complejos y su comprensión de una forma simple.

```
heatmap(matrix_felicidad, scale="column")
```

En este mapa de calor podemos ver la relación de los índices de felicidad en comparación con la libertad de elección según los países.



ces
ore

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Gráfico de puntos

Un gráfico de puntos se utiliza para representar cualquier dato en forma de puntos. Es similar a un histograma simplificado o a un gráfico de barras, ya que la altura de la barra formada con puntos representa el valor numérico de cada variable. Los gráficos de puntos se utilizan para representar pequeñas cantidades de datos. Un tipo de gráfico de puntos es el **diagrama de dispersión** (también conocido como gráfico de dispersión) utiliza puntos para representar los valores de dos variables numéricas diferentes. La posición de cada punto en el eje horizontal y vertical indica los valores de un punto de datos individual.

#Preparar un área para trazar dos gráficos

```
par(mar=c(5, 1, 4, 18), xpd=TRUE) #puede variar
```

#Generación del gráfico

```
plot(happines$Freedom.to.make.life.choices, happines$Ladder.score,  
col = as.factor(happines$Regional.indicator), #Colores  
pch = 20, #diferentes formas  
cex = 1) #tamaño del punto
```

Añade el subtítulo

```
legend("topright", inset=c(-0.7, 0), legend=unique(happines$Regional.indicator), pch = 20,  
col = as.factor(unique(happines$Regional.indicator)), cex = 0.7)
```

¡Manos a la obra!

Cree un gráfico de puntos con Expectativa de vida (*Healthy Life expectancy*) por Generosidad (*Generosity*).

Guardando gráficos

```
png("mi_grafico.png")

#Preparar un área para trazar dos gráficos
par(mar=c(5, 1, 4, 18), xpd=TRUE) #puede variar

#Generación del gráfico
plot(happines$Freedom.to.make.life.choices, happines$Ladder.score,
col = as.factor(happines$Regional.indicator), #Colores
pch = 20, #diferentes formas
cex = 1) #tamaño del punto

# Añade el subtítulo
legend("topright", inset=c(-0.7, 0), legend=unique(happines$Regional.indicator), pch = 20,
col = as.factor(unique(happines$Regional.indicator)), cex = 0.7)

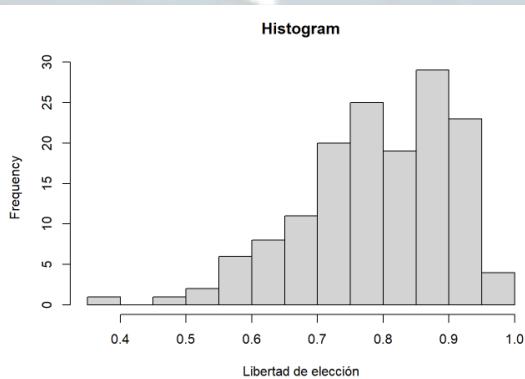
dev.off()
```

Visualización de datos

Histograma

Un histograma indica la frecuencia con que se produce cada valor diferente en un conjunto de datos. Un histograma es el gráfico más utilizado para mostrar distribuciones de frecuencia.

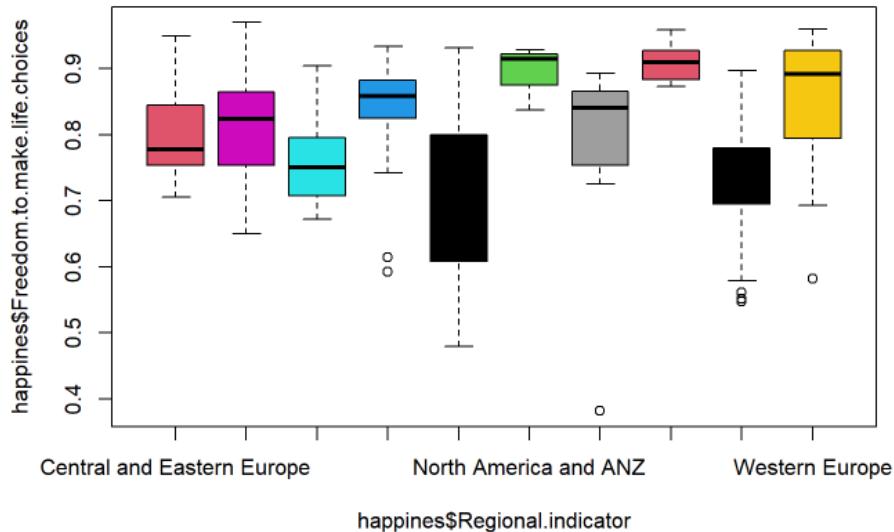
```
hist(happines$Freedom.to.make.life.choices,
     main = "Histogram", #Título
     xlab = "Libertad de elección") #Nombre del eje x
```



Gráficos de caja

El gráfico de caja es una herramienta gráfica que permite visualizar la distribución y los valores atípicos de los datos, proporcionando así un medio complementario para desarrollar el carácter de los datos. Además, el gráfico de caja es también un esquema gráfico comparativo.

```
boxplot(happines$Freedom.to.make.life.choices ~ happines$Regional.indicator,
        col = as.factor(unique(happines$Regional.indicator)))
```

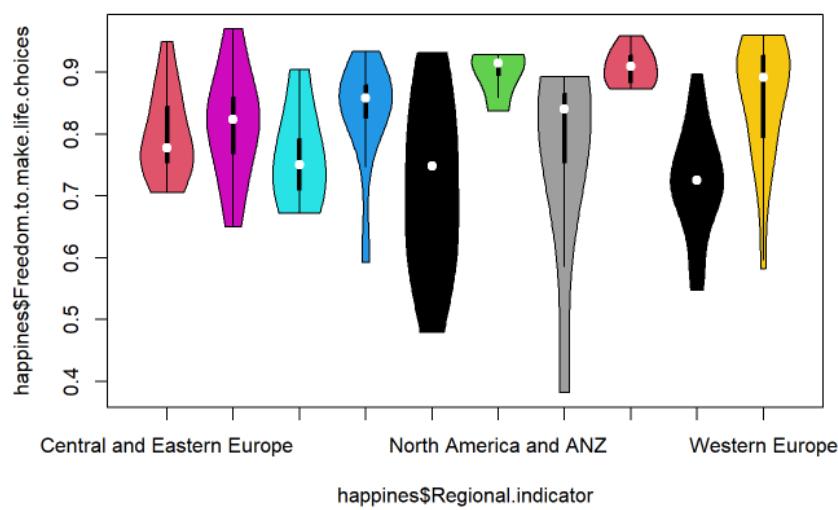


Gráficos de violín

Los gráficos de violín son similares a los gráficos de caja, excepto que también muestran la densidad de probabilidad del conjunto de los datos en diferentes valores. Generalmente, los gráficos de violín incluyen un marcador para la mediana de los datos y un recuadro que indica el rango inter cuartil, como en los gráficos de caja estándar.

```
#install.packages("vioplot")
library(vioplot)
```

```
vioplot(happines$Freedom.to.make.life.choices ~ happines$Regional.indicator,
        col = as.factor(unique(happines$Regional.indicator)))
```



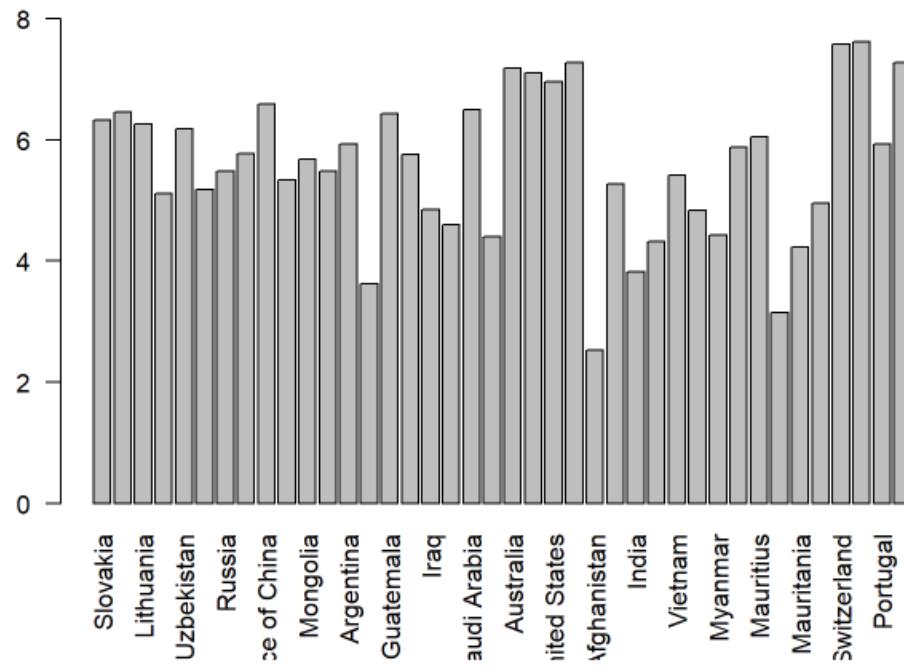
Visualización de datos 2

En este paso utilizaremos un subgrupo de los datos utilizados en la tercera clase. Para obtenerlos haga [clic aquí](#).

Gráfico de barras

```
felicidad = read.csv("felicidad_red.csv")
```

```
barplot(felicidad$Indice_felicidad, ylim=c(0,8), names.arg=felicidad$Pais, las=2)
```



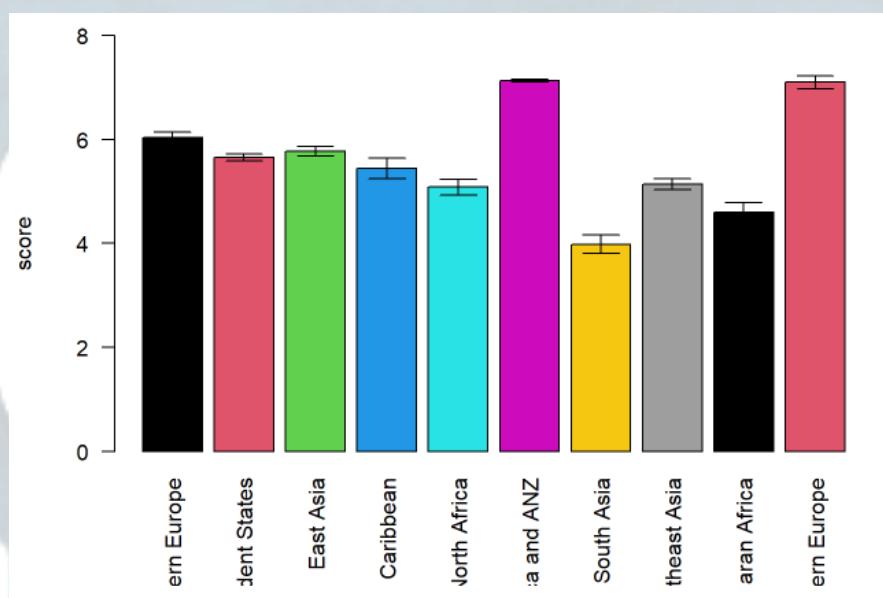
```
regiones = c()
```

```
regiones$names = unique(felicidad$Region)
```

```
for (i in 1:length(regiones$names)){
  print(regiones$names[i])
  regiones$media[i] = mean(felicidad$Indice_felicidad[felicidad$Region ==
    regiones$names[i] ])
  regiones$erro[i] = sd(felicidad$Indice_felicidad[felicidad$Region == regiones$names[i] ])/sqrt(length(felicidad$Indice_felicidad))
}
```

```
## [1] "Central and Eastern Europe"  
## [1] "Commonwealth of Independent States"  
## [1] "East Asia"  
## [1] "Latin America and Caribbean"  
## [1] "Middle East and North Africa"  
## [1] "North America and ANZ"  
## [1] "South Asia"  
## [1] "Southeast Asia"  
## [1] "Sub-Saharan Africa"  
## [1] "Western Europe"
```

```
x = barplot(regiones$media,beside=T,ylim=c(0,8),ylab="score",  
            names.arg = regiones$names,  
            las = 2,  
            col = as.factor(unique(regiones$names)))  
  
arrows(x0=x,y0=regiones$media-regiones$erro,  
       x1=x,y1=regiones$media+regiones$erro,  
       angle=90,length=0.14,code=3)
```



Histograma

Un histograma indica la frecuencia con que se produce cada valor diferente en un conjunto de datos. Un histograma es el gráfico más utilizado para mostrar distribuciones de frecuencia.

```
hist(felicidad$Indice_felicidad, main = "Histograma",
      xlab = "Indice felicidad")
```

Histograma

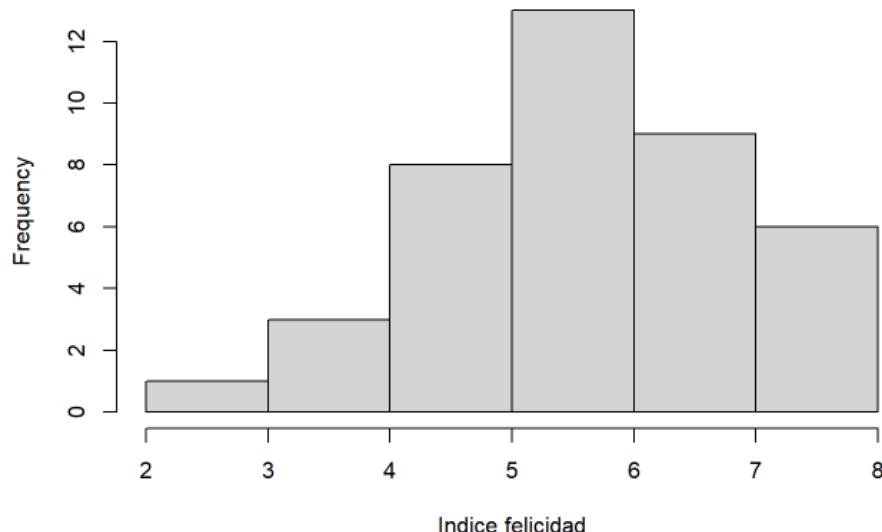
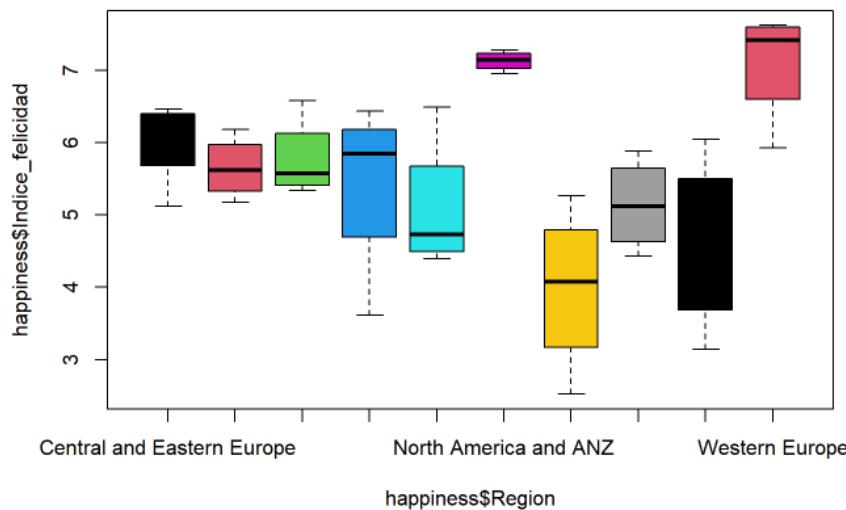


Diagrama de caja

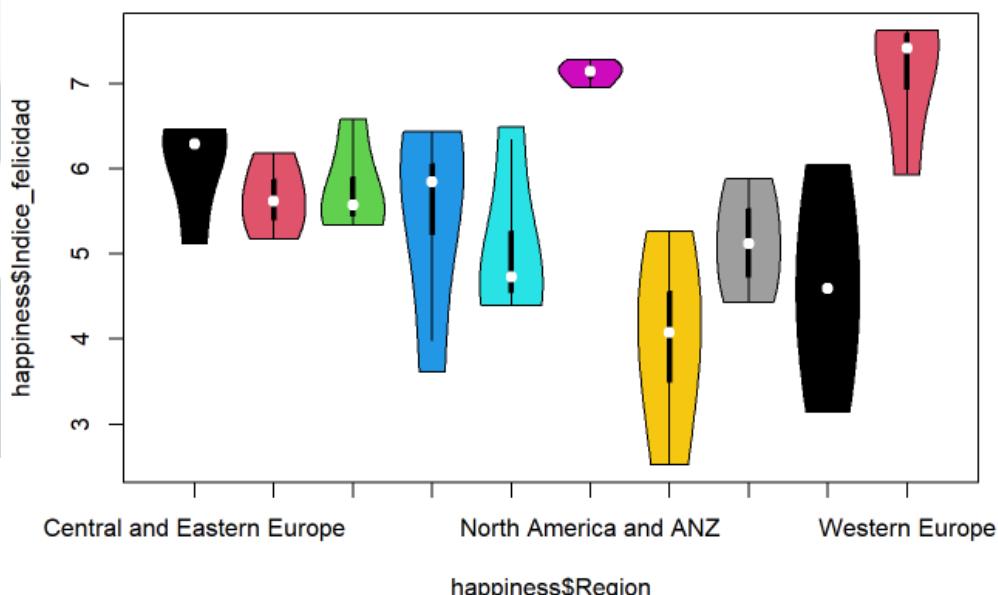
El diagrama de caja es una herramienta gráfica que permite visualizar la distribución y los valores atípicos de los datos, proporcionando así un medio complementario para desarrollar el carácter de los datos. Además, el gráfico de caja es también un esquema gráfico comparativo.

```
boxplot(felicidad$Indice_felicidad ~ felicidad$Region,  
        col = as.factor(unique(felicidad$Region)))
```



```
library(vioplot)
```

```
vioplot(felicidad$Indice_felicidad ~ felicidad$Region,
        col = as.factor(unique(felicidad$Region)))
```



Estadística descriptiva básica

Vamos evaluar la variable Indice_felicidad:

Podemos ordenar la tabla por el indice de felicidad:

```
felicidad[order(felicidad$Indice_felicidad),]
```

	Pais	Region
## 25	Afghanistan	South Asia
## 34	Zimbabwe	Sub-Saharan Africa
## 14	Haiti	Latin America and Caribbean
## 27	India	South Asia
## 35	Mauritania	Sub-Saharan Africa
## 28	Sri Lanka	South Asia
## 20	Jordan	Middle East and North Africa
## 31	Myanmar	Southeast Asia
## 18	Tunisia	Middle East and North Africa
## 30	Cambodia	Southeast Asia
## 17	Iraq	Middle East and North Africa
## 36	South Africa	Sub-Saharan Africa
## 4	Albania	Central and Eastern Europe
## 6	Azerbaijan	Commonwealth of Independent States
## 26	Nepal	South Asia
## 10	China	East Asia
## 29	Vietnam	Southeast Asia
## 7	Russia	Commonwealth of Independent States
## 12	Hong Kong S.A.R. of China	East Asia
## 11	Mongolia	East Asia
## 16	Ecuador	Latin America and Caribbean
## 8	Moldova	Commonwealth of Independent States
## 32	Philippines	Southeast Asia
## 13	Argentina	Latin America and Caribbean
## 39	Portugal	Western Europe
## 33	Mauritius	Sub-Saharan Africa
## 5	Uzbekistan	Commonwealth of Independent States
## 3	Lithuania	Central and Eastern Europe
## 1	Slovakia	Central and Eastern Europe
## 15	Guatemala	Latin America and Caribbean
## 2	Slovenia	Central and Eastern Europe
## 19	Saudi Arabia	Middle East and North Africa
## 9	Taiwan Province of China	East Asia
## 23	United States	North America and ANZ
## 22	Canada	North America and ANZ
## 21	Australia	North America and ANZ
## 40	Austria	Western Europe
## 24	New Zealand	North America and ANZ
## 37	Switzerland	Western Europe
## 38	Denmark	Western Europe
## Indice_felicidad Suporte_social Expectativa_de_vida Liberdad_de_escocas		
## 25	2.523	0.463
## 34	3.145	0.750
## 14	3.615	0.540
		52.493
		56.201
		55.700
		0.382
		0.677
		0.593

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

## 27	3.819	0.603	60.633	0.893
## 35	4.227	0.795	57.161	0.561
## 28	4.325	0.827	67.299	0.841
## 20	4.395	0.767	67.000	0.755
## 31	4.426	0.779	59.302	0.876
## 18	4.596	0.691	67.201	0.656
## 30	4.830	0.765	62.000	0.959
## 17	4.854	0.746	60.583	0.630
## 36	4.956	0.860	56.904	0.749
## 4	5.117	0.697	68.999	0.785
## 6	5.171	0.836	65.656	0.814
## 26	5.269	0.774	64.233	0.782
## 10	5.339	0.811	69.593	0.904
## 29	5.411	0.850	68.034	0.940
## 7	5.477	0.903	64.703	0.718
## 12	5.477	0.836	76.820	0.717
## 11	5.677	0.935	62.500	0.708
## 16	5.764	0.821	68.800	0.842
## 8	5.766	0.857	65.699	0.822
## 32	5.880	0.830	62.000	0.917
## 13	5.929	0.898	69.000	0.828
## 39	5.929	0.879	72.600	0.892
## 33	6.049	0.905	66.701	0.867
## 5	6.179	0.918	65.255	0.970
## 3	6.255	0.935	67.906	0.773
## 1	6.331	0.936	69.201	0.766
## 15	6.435	0.813	64.958	0.906
## 2	6.461	0.948	71.400	0.949
## 19	6.494	0.891	66.603	0.877
## 9	6.584	0.898	69.600	0.784
## 23	6.951	0.920	68.200	0.837
## 22	7.103	0.926	73.800	0.915
## 21	7.183	0.940	73.900	0.914
## 40	7.268	0.934	73.300	0.908
## 24	7.277	0.948	73.400	0.929
## 37	7.571	0.942	74.400	0.919
## 38	7.620	0.954	72.700	0.946
## Generosidad Percepcion_de_corrupcion				
## 25	-0.102	0.924		
## 34	-0.047	0.821		
## 14	0.422	0.721		
## 27	0.089	0.774		
## 35	-0.106	0.731		
## 28	0.079	0.863		
## 20	-0.167	0.705		
## 31	0.509	0.660		
## 18	-0.201	0.870		
## 30	0.034	0.843		
## 17	-0.053	0.875		
## 36	-0.067	0.860		
## 4	-0.030	0.901		
## 6	-0.223	0.506		
## 26	0.152	0.727		
## 10	-0.146	0.755		
## 29	-0.098	0.796		
## 7	-0.111	0.845		
## 12	0.067	0.403		

"INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR"

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

```
## 11   0.116    0.856
## 16  -0.124    0.843
## 8   -0.079    0.918
## 32  -0.097    0.742

## 13  -0.182    0.834
## 39  -0.244    0.887
## 33  -0.054    0.789
## 5   0.311     0.515
## 3   -0.203    0.826
## 1   -0.124    0.911
## 15  -0.038    0.775
## 2   -0.101    0.806
## 19  -0.149    0.684
## 9   -0.070    0.721
## 23  0.098     0.698
## 22  0.089     0.415
## 21  0.159     0.442
## 40  0.042     0.481
## 24  0.134     0.242
## 37  0.025     0.292
## 38  0.030     0.179
```

```
dim(felicidad)
```

```
## [1] 40 8
```

Resumen estadístico

Media

```
mean(felicidad$Indice_felicidad)
```

```
## [1] 5.59195
```

```
##   Pais      Region     Indice_felicidad Suporte_social
```

```
median(felicidad$Indice_felicidad)
```

```
## [1] 5.7205
```

```
##               Mean :5.592  Mean :0.8330
```

```
##             3rd Qu.:6.441  3rd Qu.:0.9215
```

```
##             Max. :7.620  Max. :0.9540
```

```
## Expectativa_de_vida Liberdad_de_escocjas Generosidad
```

```
## Min. :52.49  Min. :0.3820  Min. :-0.2440
```

```
## 1st Qu.:62.38  1st Qu.:0.7535  1st Qu.:-0.1143
```

```
## Median :67.10  Median :0.8390  Median :-0.0535
```

```
## Mean :66.31  Mean :0.8125  Mean :-0.0115
```

```
## 3rd Qu.:69.59  3rd Qu.:0.9095  3rd Qu.: 0.0815
```

```
## Max. :76.82  Max. :0.9700  Max. : 0.5090
```

```
## Percepcion_de_corrupcion
```

```
## Min. :0.1790
```

```
## 1st Qu.:0.6780
```

```
## Median :0.7745
```

```
## Mean :0.7109
```

```
## 3rd Qu.:0.8478
```

```
## Max. :0.9240
```

Mediana

Varianza

```
var(felicidad$Indice_felicidad)
```

```
## [1] 1.471543
```

Máximo y Mínimo

```
range(felicidad$Indice_felicidad)
```

```
## [1] 2.523 7.620
```

La media de felicidad de todas las regiones con la función aggregate()

```
aggregate(Indice_felicidad ~ Region, data = felicidad, FUN = mean)
```

##	Region	Indice_felicidad
## 1	Central and Eastern Europe	0.6218446
aggregate(Indice_felicidad ~ Region, data = felicidad, FUN = sd)		
##	Region	Indice_felicidad
## 1	Central and Eastern Europe	0.6218446
## 2	Commonwealth of Independent States	0.4292065
## 3	East Asia	0.5606106
## 4	Latin America and Caribbean	1.2469529
## 5	Middle East and North Africa	0.9580993
## 6	North America and ANZ	0.1380568
## 7	South Asia	1.1444434
## 8	Southeast Asia	0.6395036
## 9	Sub-Saharan Africa	1.2223418
## 10	Western Europe	0.7940760

El desviación standard de felicidad de todas las regiones con la función aggregate()

Prueba de hipótesis

Una hipótesis es una suposición que se hace sobre algo en una condición. Generalmente, se refiere a un hecho común o a un hecho ya aceptado.

La hipótesis nula se refiere a que cualquier cambio que se produzca en un objeto de estudio se debe al azar y no hay ningún efecto específico. Por lo tanto, se considera, por ejemplo, que los medios serán iguales, aunque las condiciones sean diferentes.

La hipótesis alternativa indica que el cambio en el objeto de estudio puede deberse a alguna condición específica y no solo al azar. Por tanto, la hipótesis alternativa considera que la media del fenómeno en la condición 1 es diferente de la media del fenómeno en la condición 2.

Las pruebas de hipótesis nos permiten realizar comparaciones para tomar la decisión de rechazar o no la hipótesis nula y decir si una alteración que se produce en nuestro objeto de estudio se debe al azar o si hay algún factor específico que influye en estas alteraciones. .

Evaluando la normalidad de los datos

```
shapiro.test(felicidad$Indice_felicidad) # (Shapiro-Wilk)
```



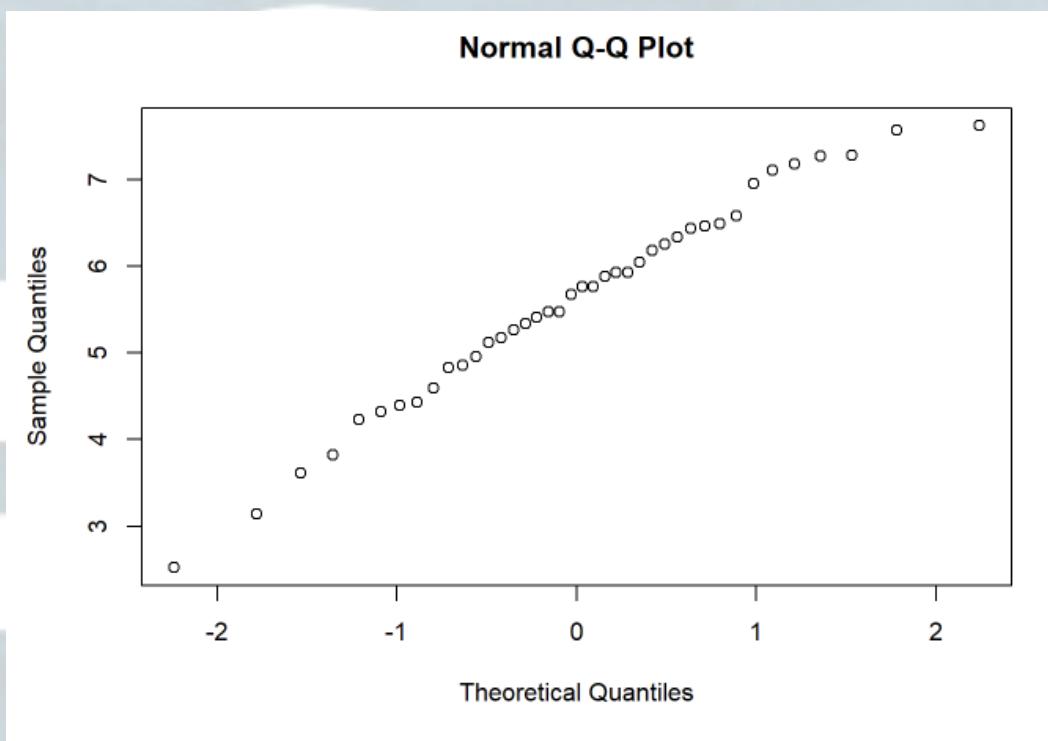
“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

```
##  
## Shapiro-Wilk normality test  
##  
## data: felicidad$Indice_felicidad  
## W = 0.97915, p-value = 0.658
```

```
qqnorm(felicidad$Indice_felicidad) # Q-Q plot
```



Intervalo de confianza

```
ic_95_med = t.test(felicidad$Indice_felicidad, conf.level=.95)  
ic_95_med
```

```
##  
## One Sample t-test  
##  
## data: felicidad$Indice_felicidad  
## t = 29.155, df = 39, p-value < 2.2e-16  
## alternative hypothesis: true mean is not equal to 0  
## 95 percent confidence interval:  
## 5.203991 5.979909  
## sample estimates:  
## mean of x  
## 5.59195
```

Análisis de varianza

Existen varios métodos para comprobar las hipótesis. Aquí hablaremos del análisis de la varianza. Con este análisis es posible definir qué variables tienen un efecto significativo sobre el objeto de estudio. La decisión de rechazar o no la hipótesis nula, se hace con base en un valor *p* comparado con un valor de confianza, que se llama alfa.

El análisis de varianza permite comparar varios grupos al mismo tiempo para un mismo efecto y asume que el azar solo produce pequeñas desviaciones, mientras que las grandes diferencias son generadas por causas reales.

Suposiciones del análisis de varianza

- Las variables son independientes entre si
- Las distribuciones de cada variable (y de los errores) se aproxima de una distribución normal
- Las varianzas de cada variable son aproximadamente iguales
- La variable dependiente es continua

Comprobaremos la hipótesis nula de que todas las regiones del mundo tienen índices de felicidad similares, es decir, que cualquier variación en el índice se debe al azar y no al efecto de la región en la que se encuentra un país.

Creando un modelo matemático

```
modelo = lm(felicidad$Indice_felicidad ~ felicidad$Region)
```

```
modelo
```

```
##  
## Call:  
## lm(formula = felicidad$Indice_felicidad ~ felicidad$Region)  
##  
## Coefficients:  
##                 (Intercept)  
##                         6.0410  
## felicidad$RegionCommonwealth of Independent States  
##                         -0.3928  
## felicidad$RegionEast Asia  
##                         -0.2718  
## felicidad$RegionLatin America and Caribbean  
##                         -0.6053  
## felicidad$RegionMiddle East and North Africa  
##                         -0.9562  
## felicidad$RegionNorth America and ANZ  
##                         1.0875  
## felicidad$RegionSouth Asia  
##                         -2.0570  
## felicidad$RegionSoutheast Asia  
##                         -0.9043  
## felicidad$RegionSub-Saharan Africa  
##                         -1.4468  
## felicidad$RegionWestern Europe  
##                         1.0560
```

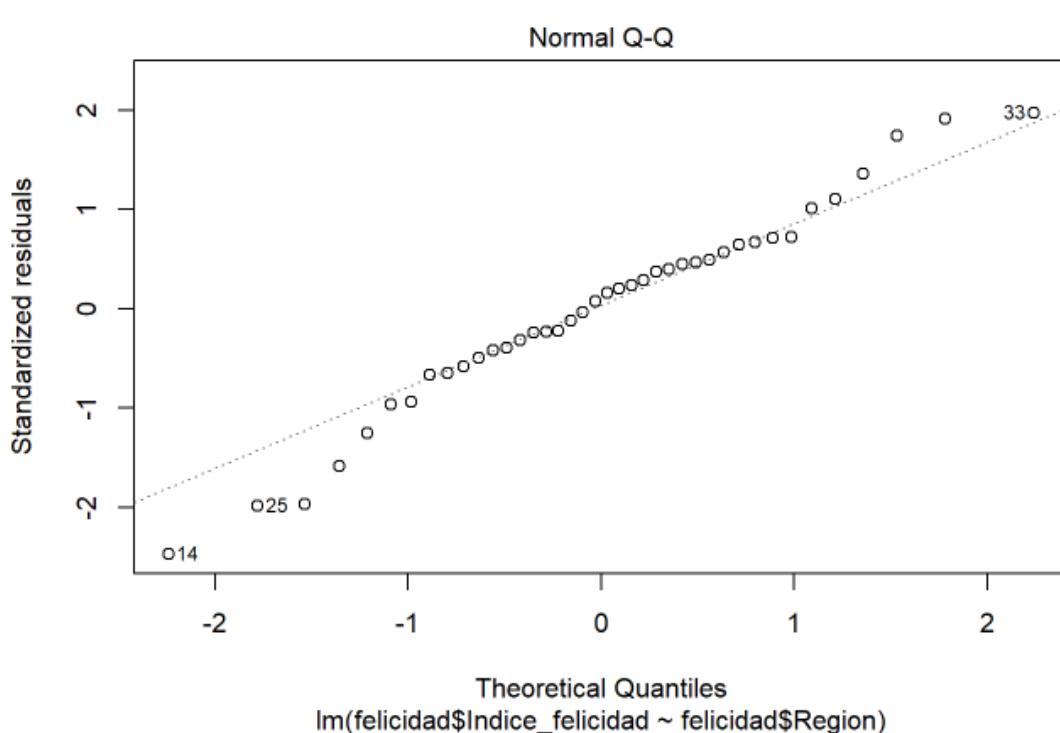
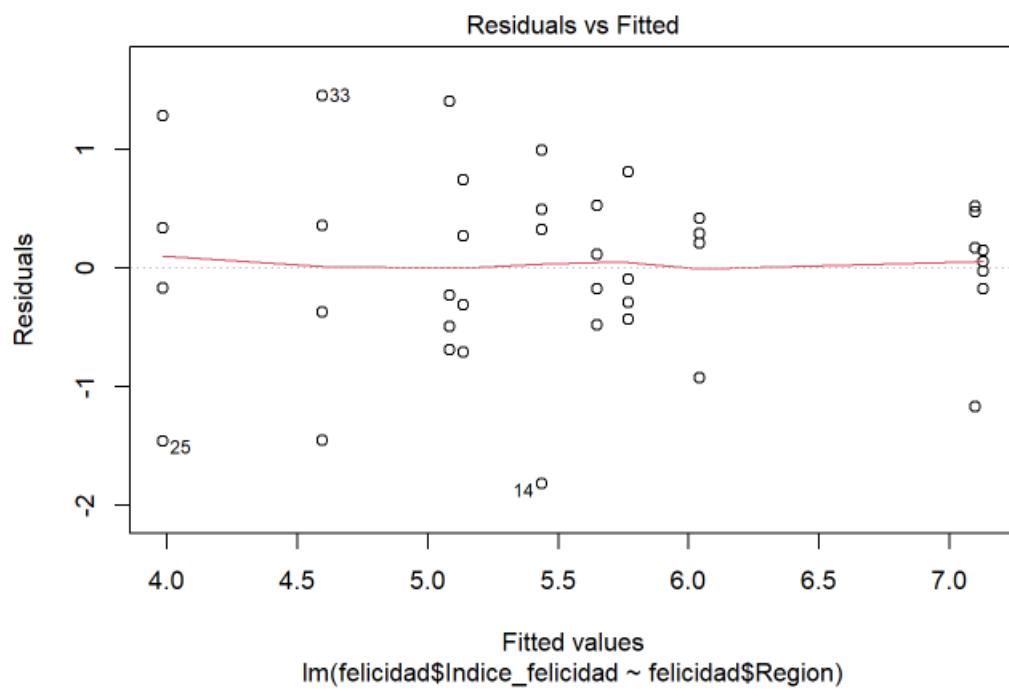
“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Diagnóstico del modelo

```
plot(modelo)
```

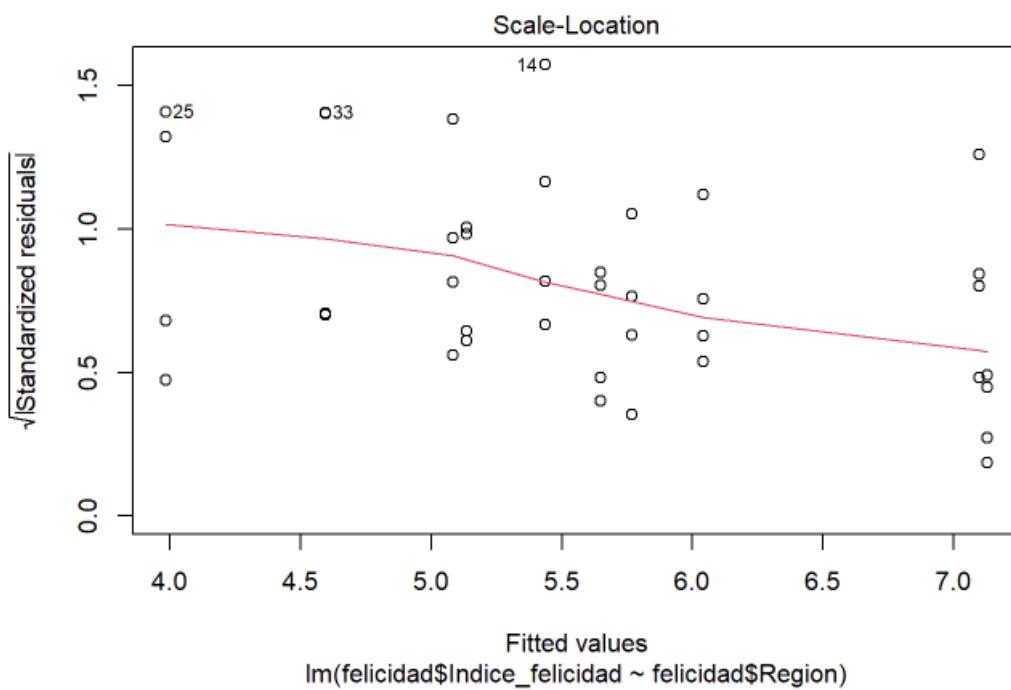


“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

```
## hat values (leverages) are all = 0.25
## and there are no factor predictors; no plot no. 5
```



El análisis de varianza

```
anova(modelo)

## Analysis of Variance Table

##
## Response: felicidad$Indice_felicidad
##          Df Sum Sq Mean Sq F value    Pr(>F)
## felicidad$Region  9 35.729  3.9699  5.498 0.0001786 ***
## Residuals     30 21.661  0.7220
## ---
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '' 1
```

Prueba de medios

Para hacer la prueba de medios utilizando el test de Tukey, vamos utilizar una función del paquete agricolae. Para eso, instale el paquete con la función `install.packages()`

```
library(agricolae)
```

```
tukey = HSD.test(modelo, 'felicidad$Region', alpha = 0.05)  
tukey
```

¡Manos a la obra!

Haga un análisis de varianza y una prueba de medios para la variable expectativa de vida. ¿Cuál es su conclusión sobre los resultados?

```

## $statistics

## MSerror Df  Mean   CV  MSD
## 0.722049 30 5.59195 15.19568 2.04962

##
## $parameters

## test      name.t ntr StudentizedRange alpha
## Tukey felicidad$Region 10    4.824141 0.05

##
## $means

##           felicidad$Indice_felicidad   std r  Min
## Central and Eastern Europe       6.04100 0.6218446 4 5.117
## Commonwealth of Independent States      5.64825 0.4292065 4 5.171
## East Asia                      5.76925 0.5606106 4 5.339
## Latin America and Caribbean      5.43575 1.2469529 4 3.615
## Middle East and North Africa     5.08475 0.9580993 4 4.395
## North America and ANZ          7.12850 0.1380568 4 6.951
## South Asia                     3.98400 1.1444434 4 2.523
## Southeast Asia                  5.13675 0.6395036 4 4.426
## Sub-Saharan Africa              4.59425 1.2223418 4 3.145
## Western Europe                 7.09700 0.7940760 4 5.929
##
##           Max   Q25   Q50   Q75
## Central and Eastern Europe     6.461 5.97050 6.2930 6.36350
## Commonwealth of Independent States 6.179 5.40050 5.6215 5.86925
## East Asia                     6.584 5.44250 5.5770 5.90375
## Latin America and Caribbean    6.435 5.22675 5.8465 6.05550
## Middle East and North Africa    6.494 4.54575 4.7250 5.26400
## North America and ANZ         7.277 7.06500 7.1430 7.20650
## South Asia                    5.269 3.49500 4.0720 4.56100
## Southeast Asia                 5.880 4.72900 5.1205 5.52825
## Sub-Saharan Africa             6.049 3.95650 4.5915 5.22925
## Western Europe                 7.620 6.93325 7.4195 7.58325
##
## $comparison
## NULL
##
## $groups

##           felicidad$Indice_felicidad groups
## North America and ANZ          7.12850   a
## Western Europe                  7.09700   a
## Central and Eastern Europe      6.04100   ab
## East Asia                      5.76925   abc
## Commonwealth of Independent States 5.64825   abc
## Latin America and Caribbean    5.43575   abc
## Southeast Asia                  5.13675   abc
## Middle East and North Africa    5.08475   abc
## Sub-Saharan Africa              4.59425   bc
## South Asia                     3.98400   c
##
## attr(,"class")
## [1] "group"

```

Actividades prácticas

En esta actividad práctica utilizaremos las funciones discutidas el primer día del curso.

Consejos:

1. Cree un archivo Rmarkdown llamado “Actividad Práctica Día 1” con salida HTML. Actualice el título, su nombre y la fecha del documento. El primer día describimos cómo crear un archivo Rmarkdown, al que se puede acceder [aquí](#).
2. Recuerde que para los caracteres (letras) se utilizan comillas.
3. Ejecute los códigos R dentro de las porciones de código R. Fuera de ellos describe la actividad o la respuesta a las preguntas.

Evaluaremos principalmente el desarrollo del script y el resultado final.

[Sube aquí los archivos de la primera actividad práctica.](#)

Preguntas:

1. Haz un vector de secuencias de 0 hasta 5, con intervalo de 0.5 en 0.5 y guarda este vector en un objeto llamado **mi_seq**. ¿Cuál es la clase (o tipo) del objeto **mi_seq**?
2. Ahora crea un nuevo objeto que almacene una repetición de tres veces de los valores dentro del objeto **mi_seq**. Elije el nombre que creas más apropiado para tu objeto.
3. ¿Cuál es la suma de los valores del objeto do ejercicio anterior? ¿Cuál es el valor medio del objeto?
4. Multiplica el vector **mi_seq** y el vector que has creado. ¿Es adecuada esta multiplicación?
5. Ahora multiplica el vector **mi_seq** por el vector **c(2,4)**? ¿Qué pasa?
6. Crea un objeto llamado **mi_casa** y almacene los nombres de las personas que viven en tu casa. ¿Cuál es la estructura del objeto **mi_casa**?
7. Multiplica el vector **mi_casa** por 2. ¿Qué pasa? ¿Por qué?
8. El día tiene 1440 minutos. Si dedico 112 minutos a resolver estos ejercicios, ¿Cuántas horas me quedan al día? (Sugerencia, divide el resultado inicial entre 60).

Si tienes alguna duda, no dudes en preguntar en cualquiera de nuestras plataformas, de preferencia por [Google classroom](#).

Actividades prácticas

Consejos:

1. Cree un archivo Rmarkdown llamado “Actividad Práctica Día 2” con salida HTML. Actualice el título, su nombre y la fecha del documento. El primer día describimos cómo crear un archivo Rmarkdown, al que se puede acceder [aquí](#).
2. Ejecute los códigos R dentro de las porciones de código R (chuncks). Fuera de ellos describe la actividad o la respuesta a las preguntas.
3. Recuerde de las funciones de importación de datos. (Ex. read.csv). El segundo día describimos cómo importar un archivo de texto, al que se puede acceder [aquí](#).

Obs. Hay un parámetro de read.csv que necesitas utilizar para reconocer los caracteres especiales latinos (encoding = "UTF-8"). Ex. read.csv (file = "nombre_del_archivo", encoding = "UTF-8")

Evaluaremos principalmente **el desarrollo del script** y el resultado final.

[Sube aquí los archivos de la según actividad práctica.](#)

En la actividad práctica de hoy exploraremos un poco los datos del relevamiento de especies por clases de animales de la Amazonía peruana realizado en el periodo de 2010 a 2019. En este ejercicio discutiremos algunos **conceptos abordados en la lección del día 2** para facilitar la comprensión de los datos. Para descargar los datos haga [clic aquí](#).

Categoría	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
Anfibios	538	538	538	538	624	588	588	603	622	626
Reptiles	400	421	421	446	446	452	455	467	469	474
Aves	1 831	1 835	1 835	1 847	1 847	1 849	1 852	1 857	1 857	1 857
Mamíferos	508	512	519	524	525	537	545	551	559	559
Peces continentales	855	1 010	1 064	1 064	1 064	1 064	1 064	1 064	1 141	1 141
Peces marinos	1 090
Angiospermas y gimnospermas	19 000	19 000	19 000	19 000	19 000	19 000	19 000	19 147	19 174	19 422

Preguntas:

1. ¿Cuál es la estructura de los datos del archivo `tarea_actividades_practicas_dia_2.csv`?
2. ¿Qué tipos de variables puedes identificar? ¿Qué quiere decir con cada uno de ellos?
3. ¿Cuántas variables son de tipo de character?
4. ¿Cuál es el número total de especies por año?

Resultado esperado:

5. ¿Cuál es la diferencia en el número de especies de peces continentales entre los años 2010 y 2019?

Resultado esperado:

6. ¿Cuántas especies tienen más aves en comparación con los mamíferos en 2019?

Resultado esperado:

7. Porcentaje de los números de especies respectivas para el año 2019.

Resultado esperado:

8. ¿Cuál es la razón del número total de especies para 2019 y 2010?

Resultado esperado:

9. ¿Es el número de especies de aves en 2016 mayor que la suma del número de peces de 2012 a 2016? (Lógico)

Resultado esperado:

Actividades prácticas

Consejos:

1. Cree un archivo Rmarkdown llamado “Actividad Práctica Día 3” con salida HTML. Actualice el título, su nombre y la fecha del documento. El primer día describimos cómo crear un archivo Rmarkdown, al que se puede acceder [aquí](#).
2. Ejecute los códigos R dentro de las porciones de código R (chuncks). Fuera de ellos describe la actividad o la respuesta a las preguntas.
3. Recuerde de las funciones de importación de datos. (Ex. read.csv). El segundo día describimos cómo importar un archivo de texto, al que se puede acceder [aquí](#).

Evaluaremos principalmente **el desarrollo del script** y el resultado final.

[Sube aquí los archivos de la tercera actividad práctica.](#)

En esta actividad utilizaremos un conjunto de datos que evalúan el consumo de energía en el mundo por continentes. En este ejercicio discutiremos algunos conceptos abordados en la lección para facilitar la comprensión de los datos. Para descargar los datos [haga clic aquí](#).

Year	World	Europe	North America	Latin America	Asia	Africa
1990	101855.54	20654.88	24667.23	5373.06	24574.19	4407.77
1991	102483.56	20631.62	24841.68	5500.99	24783.53	4535.7
1992	102588.23	20189.68	25341.77	5628.92	25690.67	4582.22
1993	103646.56	20189.68	25830.23	5675.44	26876.93	4721.78
1994	104449.03	20085.01	26365.21	5989.45	28098.08	4803.19
1995	107112.3	20713.03	26714.11	6024.34	29761.17	5000.9
1996	109763.94	21445.72	27295.61	6303.46	30772.98	5152.09
1997	110903.68	21341.05	27574.73	6570.95	31435.89	5280.02
1998	111450.29	21503.87	27772.44	6803.55	31331.22	5431.21
1999	113974	21306.16	28528.39	6989.63	32412.81	5559.14

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Year	World	Europe	North America	Latin America	Asia	Africa
2000	116590.75	21538.76	29342.49	7024.52	33564.18	5617.29
2001	117521.15	21934.18	28795.88	7036.15	34227.09	5756.85
2002	120207.68	21969.07	29156.41	7280.38	35680.84	5849.89
2003	124464.26	22515.68	29377.38	7501.35	38181.29	6140.64
2004	129953.62	22748.28	29993.77	7780.47	41437.69	6466.28
2005	133582.18	22864.58	30168.22	8059.59	43693.91	6652.36
2006	137396.82	23108.81	29958.88	8303.82	46124.58	6861.7
2007	141211.46	22829.69	30424.08	8571.31	48555.25	7129.19
2008	142874.55	22829.69	29644.87	8780.65	49636.84	7443.2
2009	141490.58	21608.54	28214.38	8559.68	51858.17	7606.02
2010	149294.31	22469.16	28830.77	9117.92	56010.08	7803.73
2011	151783.13	21794.62	28598.17	9304	58324.45	8117.74
2012	153748.6	21689.95	28167.86	9594.75	60022.43	8303.82
2013	155958.3	21480.61	28667.95	9978.54	61499.44	8431.75
2014	157667.91	20689.77	29063.37	9943.65	62895.04	8896.95
2015	158086.59	21038.67	28784.25	9873.87	63476.54	8838.8
2016	159377.52	21224.75	28563.28	9745.94	63941.74	9106.29
2017	162459.47	21620.17	28563.28	9687.79	65721.13	9304
2018	166297.37	21399.2	29470.42	9513.34	68035.5	9559.86
2019	167553.41	21061.93	29249.45	9478.45	69582.29	9641.27
2020	160819.64	19643.07	27063.01	8815.54	69256.65	9408.67

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>

Preguntas:

1. Importe la tabla de datos utilizando la función `read.csv()` y evalúe la estructura de los datos. Selecciona solo las columnas Año (Year), Mundo (World), Europa, América del Norte, América Latina, Asia y África y almacena en un nuevo objeto.

2. Crea un gráfico de barras que muestre el consumo mundial de energía desde 1990 hasta 2020.

Nota: No olvides poner los años y girar 90 grados para que aparezcan todos los años.

Desafío: Haga un gráfico de líneas con la misma información arriba. (Consejo: use el? `plot` para buscar ayuda para transformar el gráfico en tipo "línea").

3. Crea el gráfico de barras de 1999 a 2020 considerando únicamente América Latina.

4. Haga un heatmap con el consumo de energía en los países en relación al consumo de los continentes. Acuérdate de hacer una matriz con los datos que se desea hacer el gráfico (función `as.matrix`). ¿Qué continentes consumieron más energía (más rojo) y qué continentes consumieron menos energía (más amarillo) en el año 2003?

5. Haga un gráfico de puntos relacionando el consumo de energía en América latina y el consumo mundial.

6. Hace un gráfico con el consumo de energía por región en 2020. (Si desea colorear las barras, puede utilizar los nombres de las columnas de la matriz mediante la función de `names()`).

7. Seleccione las columnas año y América latina y almacénelas en un objeto llamado `energy_latam`. Tomar el consumo medio de los años y almacenarlo en un objeto llamado `media`. Haz un loop condicional para que se ejecute en el número de filas. En este loop cree una condición para comprobar si el valor es mayor o menor que la media. Si el valor es mayor que la media escribe "Consumimos mucha energía" en una nueva columna llamada `consumo`, en caso contrario escribe "Consumimos poca energía" en la misma columna.

Si tienes alguna duda o consulta sobre este curso no dudes en visitar nuestras redes sociales oficiales:

<https://web.facebook.com/CIICAMPERU>

<https://web.facebook.com/ciicamresearchcenter>

<https://www.instagram.com/ciicamresearchcenter>

https://www.youtube.com/channel/UC29_KUXOQ8e6g6xUcLF_FNQ

<https://twitter.com/CiicamC>

<https://www.linkedin.com/company/69539004>

¡Hasta una nueva edición, esperamos verte en uno de nuestros cursos!

LA FAMILIA CIICANENSE.

“INVESTIGAR PARA CONOCER, CONOCER PARA TRANSFORMAR”

www.ciicam.com

<https://ciicam.github.io/cursoR/index.html>