Guía Completa de R y Estadística para Principiantes

Un Manual Práctico para el Análisis de Datos desde Cero

Índice

6. **Capítulo 6: Importar y Exportar Datos**
    - 6.1. Importar Datos desde un archivo CSV
    - 6.2. Exportar Datos a un archivo CSV
7. **Capítulo 7: Estadística Descriptiva**
    - 7.1. Medidas de Tendencia Central
    - 7.2. Medidas de Dispersión
    - 7.3. Correlación
    - 7.4. Ejercicios Prácticos del Capítulo 7
8. **Capítulo 8: Visualización de Datos**
    - 8.1. Histogramas: Viendo la Distribución
    - 8.2. Gráficos de Dispersión: Encontrando Relaciones
    - 8.3. Diagramas de Caja (Boxplots): Comparando Grupos
    - 8.4. Ejercicios Prácticos del Capítulo 8
9. **Capítulo 9: Buenas Prácticas de Programación**
    - 9.1. Nombres de Variables
    - 9.2. Comentarios Efectivos
    - 9.3. Estructura y Organización del Código

Capítulo 1: Introducción a R

1.1. ¿Qué es R y por qué es tan popular?

R es un lenguaje de programación y un entorno de software diseñado específicamente para el **cálculo estadístico y la visualización de datos**. Creado por estadísticos para estadísticos, se ha convertido en la herramienta preferida por científicos de datos, analistas y académicos de todo el mundo.

Piensa en R como una calculadora extremadamente potente que no solo realiza operaciones complejas, sino que también te permite manejar grandes volúmenes de datos y crear gráficos de calidad profesional para comunicar tus hallazgos.

1.2. Ventajas de aprender R

- **Gratuito y de Código Abierto:** No necesitas pagar costosas licencias.
- **Comunidad Enorme:** Millones de usuarios comparten soluciones y paquetes.
- **Potencia Estadística:** Incluye cualquier prueba estadística que puedas imaginar.
- **Visualización de Datos Superior:** Con paquetes como ggplot2, puedes crear gráficos impresionantes.
- **Alta Demanda Laboral:** Es una de las habilidades más buscadas en el campo del análisis de datos.

Capítulo 2: Preparando el Entorno de Trabajo

2.1. Instalación de R

R es el "motor". Es lo primero que debemos instalar.

1. Ve a la página oficial de CRAN: https://cran.r-project.org/bin/windows/base/
2. Haz clic en el enlace de descarga más reciente.
3. Ejecuta el instalador y acepta todas las opciones por defecto.

2.2. Instalación de RStudio

RStudio es la "cabina del piloto". Es un Entorno de Desarrollo Integrado (IDE) que hace que usar R sea mucho más fácil y organizado.

1. Ve a la página de descargas de Posit: https://posit.co/download/rstudio-desktop/
2. Descarga la versión gratuita ("Free") para tu sistema operativo.
3. Ejecuta el instalador y sigue los pasos.

2.3. Conociendo la Interfaz de RStudio: Los 4 Paneles

Al abrir RStudio, verás cuatro ventanas principales:

1. **Editor de Scripts (Arriba Izquierda):** Tu cuaderno digital. Aquí escribes y guardas tu código para reutilizarlo.
2. **Consola (Abajo Izquierda):** El cerebro de R. Aquí se ejecutan los comandos y ves los resultados.
3. **Entorno (Arriba Derecha):** Tu inventario. Muestra todas las variables y datos que has creado.
4. **Archivos/Gráficos (Abajo Derecha):** Tu caja de herramientas. Te permite ver archivos, gráficos, paquetes y la ayuda.

Capítulo 3: Tus Primeros Pasos en R

3.1. El Directorio de Trabajo

Es la carpeta por defecto donde R buscará y guardará archivos.

Generated R

```
    # Ver el directorio actual
getwd()


# Establecer un nuevo directorio (¡usa siempre "/"!)
setwd("C:/Users/TuUsuario/Documents/Mi_Proyecto_R")


# Ver los archivos en el directorio actual
list.files()
```

3.2. Creando Variables

Las variables guardan información. Usamos <- (atajo: Alt + -) para asignar valores.

Generated R

```
    # Variable numérica
```

```
mi_edad <- 28


# Variable de texto (siempre entre comillas)
mi_nombre <- "Ana"


# Para ver el contenido de una variable, solo escribe su nombre
mi_edad
mi_nombre
```

3.3. Operaciones Matemáticas Básicas

R funciona como una calculadora.

Generated R

```
    5 + 3       # Suma
10 - 4      # Resta
6 * 7       # Multiplicación
15 / 3      # División
2 ^ 4       # Potencia
```

3.4. Tu Primer Script en R

Un script es un archivo de texto con extensión .R que contiene tu código.

1. En RStudio, ve a File > New File > R Script (o Ctrl+Shift+N).
2. Escribe tu código en la nueva ventana.
3. Selecciona las líneas que quieres ejecutar y presiona Ctrl+Enter.
4. Guarda tu trabajo con Ctrl+S.

3.5. Ejercicios Prácticos del Capítulo 3

1. **Ejercicio 1:** Crea una variable llamada año_nacimiento con tu año de nacimiento y otra llamada año_actual con el año actual. Crea una tercera variable mi_edad_calculada que reste ambas para calcular tu edad.
2. **Ejercicio 2:** Crea un script que asigne tu nombre a una variable y luego imprima en la consola el mensaje "Hola, [tu nombre]!".

Capítulo 4: Fundamentos del Lenguaje

4.1. Operadores en R

- **Aritméticos:** +, -, *, /, ^ (potencia), %% (módulo/resto).
- **De Comparación:** == (igual a), != (diferente de), >, <, >=, <=. Devuelven TRUE o FALSE.
- **Lógicos:** & (Y), | (O), ! (NO). Se usan para combinar condiciones.

Generated R

```
edad <- 20
nota <- 85

# ¿El estudiante aprobó Y es mayor de edad?
(nota >= 60) & (edad >= 18)
# [1] TRUE
```

4.2. Tipos de Datos Fundamentales

- **numeric:** Números con decimales (ej: 3.14).
- **integer:** Números enteros (ej: 10L). La L le dice a R que es un entero.
- **character:** Texto (ej: "hola").
- **logical:** Valores booleanos (TRUE o FALSE).
- **factor:** Para variables categóricas (ej: "Hombre", "Mujer").

Para saber el tipo de dato de una variable, usa la función class().

Generated R

```
x <- 15.5
class(x)
# [1] "numeric"

y <- "Estadística"
class(y)
# [1] "character"
```

4.3. Vectores: La Base de R

Un vector es una lista de elementos del mismo tipo. Se crean con la función c() (concatenar).

Generated R

```
    # Vector de calificaciones
calificaciones <- c(90, 85, 78, 92, 88)

# Funciones útiles para vectores
length(calificaciones)  # Cantidad de elementos -> 5
sum(calificaciones)     # Suma total -> 433
mean(calificaciones)    # Promedio -> 86.6
max(calificaciones)     # Valor máximo -> 92
min(calificaciones)     # Valor mínimo -> 78
sort(calificaciones)    # Ordenar de menor a mayor

# Acceder a elementos (¡R empieza a contar desde 1!)
calificaciones[1]       # Primer elemento -> 90
calificaciones[c(1, 3)] # Primer y tercer elemento -> 90 78
```

4.4. Ejercicios Prácticos del Capítulo 4

1. **Ejercicio 1:** Crea un vector con los precios de 5 productos: 15, 22.5, 18, 30, 12. Calcula el precio total y el precio promedio.
2. **Ejercicio 2:** Usando el vector de precios, crea una variable lógica que indique cuáles productos cuestan más de 20.

Capítulo 5: Manejo de Datos con DataFrames

5.1. ¿Qué es un DataFrame?

Es la estructura más importante para el análisis de datos en R. Imagina una tabla de Excel, con filas (observaciones) y columnas (variables).

5.2. Creación y Exploración de DataFrames

Generated R

```
    # Crear un DataFrame
estudiantes_df <- data.frame(
  nombre = c("Ana", "Luis", "Maria"),
  edad = c(21, 23, 22),
  carrera = c("Economía", "Ingeniería", "Medicina")
)

# Explorar el DataFrame
View(estudiantes_df)   # Ver en una nueva pestaña (V mayúscula)
head(estudiantes_df)    # Ver las primeras 6 filas
str(estudiantes_df)     # Ver la estructura (tipos de datos)
summary(estudiantes_df) # Resumen estadístico de cada columna
nrow(estudiantes_df)    # Número de filas
ncol(estudiantes_df)    # Número de columnas
```

5.3. Seleccionar y Modificar Datos

Generated R

```
    # Seleccionar una columna con $
edades <- estudiantes_df$edad

# Seleccionar filas y columnas con [fila, columna]
# Fila 1, todas las columnas
estudiantes_df[1, ]

# Todas las filas, columna 2
estudiantes_df[ , 2]
```

```
# Agregar una nueva columna
estudiantes_df$semestre <- c(5, 7, 6)

# Agregar una nueva fila
nuevo_estudiante <- data.frame(nombre="Carlos", edad=24, carrera="Derecho", semestre=8)
estudiantes_df <- rbind(estudiantes_df, nuevo_estudiante)
```

5.4. Ejercicios Prácticos del Capítulo 5

1. **Ejercicio 1:** Crea un DataFrame con información de 3 libros: título, autor y año de publicación.
2. **Ejercicio 2:** Agrega una nueva columna al DataFrame llamada es_clasico que sea TRUE si el libro fue publicado antes del año 2000.

Capítulo 6: Importar y Exportar Datos

6.1. Importar Datos desde un archivo CSV

Un archivo CSV (Valores Separados por Comas) es el formato más común para compartir datos.

Generated R

```
# Asegúrate de que el archivo "mis_datos.csv" está en tu directorio de trabajo
# Si no, usa setwd() para cambiarlo

datos <- read.csv("mis_datos.csv")
```

6.2. Exportar Datos a un archivo CSV

Puedes guardar tus DataFrames modificados en un nuevo archivo.

Generated R

```
# Guardar el DataFrame 'estudiantes_df' en un nuevo archivo
# row.names = FALSE evita que se guarde una columna extra con los números de fila
write.csv(estudiantes_df, file = "estudiantes_exportado.csv", row.names = FALSE)
```

Capítulo 7: Estadística Descriptiva

7.1. Medidas de Tendencia Central

Resumen el "centro" de tus datos.

Generated R

```
    # Usando la columna de edad de nuestro DataFrame
edades <- estudiantes_df$edad


mean(edades)   # Media o promedio
median(edades) # Mediana (el valor del medio)
```

7.2. Medidas de Dispersión

Miden qué tan "esparcidos" están tus datos.

Generated R

```
    sd(edades)     # Desviación estándar
var(edades)    # Varianza
range(edades)  # Devuelve el valor mínimo y máximo
```

7.3. Correlación

Mide la relación lineal entre dos variables numéricas (de -1 a 1).

Generated R

```
    horas_estudio <- c(5, 8, 10, 12, 15)
calificacion <- c(60, 75, 80, 88, 95)


cor(horas_estudio, calificacion)
# Un valor cercano a 1 indica una fuerte relación positiva
```

7.4. Ejercicios Prácticos del Capítulo 7

1. **Ejercicio 1:** Crea un vector con 10 estaturas. Calcula la media, mediana y desviación estándar.

2. **Ejercicio 2:** ¿Qué significa una desviación estándar pequeña vs. una grande?

Capítulo 8: Visualización de Datos

8.1. Histogramas: Viendo la Distribución

Muestran la frecuencia de valores en una variable numérica.

Generated R

```
 # Crear un histograma de las edades
hist(estudiantes_df$edad,
    main = "Distribución de Edades",
    xlab = "Edad",
    ylab = "Frecuencia",
    col = "skyblue")
```

8.2. Gráficos de Dispersión: Encontrando Relaciones

Muestran la relación entre dos variables numéricas.

Generated R

```
 plot(horas_estudio, calificacion,
    main = "Horas de Estudio vs. Calificación",
    xlab = "Horas de Estudio",
    ylab = "Calificación Final",
    pch = 19, # Tipo de punto
    col = "blue")
```

8.3. Diagramas de Caja (Boxplots): Comparando Grupos

Excelentes para ver la distribución y comparar grupos.

Generated R

```
 # Comparar las edades por carrera
boxplot(edad ~ carrera, data = estudiantes_df,
      main = "Distribución de Edades por Carrera",
```

col = c("lightblue", "lightgreen", "lightcoral"))

8.4. Ejercicios Prácticos del Capítulo 8

1. **Ejercicio 1:** Usando el vector de estaturas del ejercicio anterior, crea un histograma.
2. **Ejercicio 2:** Crea dos vectores, peso y altura, para 5 personas. Haz un gráfico de dispersión para ver su relación.

Capítulo 9: Buenas Prácticas de Programación

9.1. Nombres de Variables

- **Usa nombres descriptivos:** peso_paciente_kg es mejor que x.
- **Sé consistente:** Usa snake_case (ej: mi_variable) o camelCase (ej: miVariable), pero no los mezcles.

9.2. Comentarios Efectivos

Usa el símbolo # para añadir comentarios. Explica el **"por qué"** de tu código, no el "qué".

Generated R

```
    # MAL: Sumar 5 a x
x <- x + 5


# BIEN: Ajustar el valor base por el margen de error estándar
valor_base <- valor_base + margen_error
```

9.3. Estructura y Organización del Código

Divide tu script en secciones lógicas.

Generated R

```
    # 1. Cargar librerías ----
library(dplyr)


# 2. Cargar datos ----
datos <- read.csv("datos.csv")
```

3. Limpieza de datos ----

...

4. Análisis ----

...