

▶ TALENTO **Herramientas Básicas para el Análisis de Datos**

Juan Sebastián Galindo Lizcano

1. INTRODUCCIÓN A PYTHON

- **PYTHON**

Python es un lenguaje de programación de alto nivel, interpretado y de propósito general, conocido por su sintaxis simple y legible, lo que lo hace accesible tanto para principiantes como para expertos en programación. Desde su creación por Guido van Rossum en 1991, Python ha crecido en popularidad y se ha convertido en una de las herramientas más utilizadas en la ciencia de datos, el análisis de datos, el desarrollo web, la inteligencia artificial, y muchos otros campos.




1.1 Características Clave de Python en el Análisis de Datos:

• SIMPLICIDAD Y LEGIBILIDAD

Python es conocido por su sintaxis clara y sencilla, que facilita el aprendizaje y la escritura de código. La legibilidad del código es una prioridad en Python, lo que significa que es más fácil de entender y mantener a largo plazo.

EJEMPLO DE CÓDIGO



```
1 # Calcular la media de una lista de números
2 numeros = [10, 20, 30, 40, 50]
3 media = sum(numeros) / len(numeros)
4 print(f"La media es: {media}")
```

1.2 Versatilidad Python

Python no es solo para el análisis de datos; es un lenguaje de propósito general que puede utilizarse para una amplia variedad de aplicaciones, desde el desarrollo web con frameworks como Django y Flask, hasta la automatización de tareas y la creación de aplicaciones de escritorio. Esta versatilidad permite a los científicos de datos usar Python para todo el ciclo de vida del análisis de datos, desde la recolección y limpieza de datos, hasta la visualización y el despliegue de modelos de aprendizaje automático.

1.3 Extensa Biblioteca de Herramientas y Paquetes

Python **ofrece una amplia colección de bibliotecas** que facilitan el análisis de datos. Algunas de las más populares son:

pandas: Manipulación y análisis de datos estructurados.

numpy: Cálculos numéricos y manipulación de matrices.

matplotlib y seaborn: Creación de visualizaciones de datos.

scikit-learn: Implementación de algoritmos de aprendizaje automático.

TensorFlow y Keras: Desarrollo de modelos de aprendizaje profundo.

1.4 Ventajas de Python

Interactividad: Compatible con Jupyter Notebooks, ideal para combinar código, texto y visualizaciones.

Comunidad Activa: Amplios recursos, tutoriales y foros disponibles; continua evolución de nuevas bibliotecas.

Portabilidad y Flexibilidad: Multiplataforma; código ejecutable en Windows, macOS y Linux sin cambios significativos.

Integración con Otras Tecnologías: Fácil integración con lenguajes como C/C++ y bases de datos SQL.

Automatización: Ideal para tareas repetitivas; bibliotecas como BeautifulSoup y Selenium facilitan la recolección de datos.

Aprendizaje Automático: Líder en IA, con herramientas potentes para desarrollar modelos predictivos y sistemas de recomendación.

1.4 Ejemplo Practico Python



```
1 # EJECUTAR LOS PIP EN
2 # pip install pandas matplotlib seaborn scikit-learn
3
```



```
1 # PASO 1 Carga de Datos
2 import pandas as pd
3 datos_ventas = pd.read_csv('ventas_mensuales.csv')
```



```
1 # PASO 2 Exploración de Datos
2 print(datos_ventas.head())
3 print(datos_ventas.describe())
```

1.4 Ejemplo Practico Python



```
1  # PASO 3 Visualización de Datos
2  import matplotlib.pyplot as plt
3  import seaborn as sns
4  sns.lineplot(x='Mes', y='Ventas', data=datos_ventas)
5  plt.title('Tendencias de Ventas Mensuales')
6  plt.show()
```


1.4 Ejemplo Practico Python



```
1  # PASO 4 Modelado Predictivo
2  from sklearn.model_selection import train_test_split
3  from sklearn.linear_model import LinearRegression
4
5  X = datos_ventas[['Mes']]
6  y = datos_ventas['Ventas']
7  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
8
9  modelo = LinearRegression()
10 modelo.fit(X_train, y_train)
11 prediccion = modelo.predict([[13]]) # Predicción para el mes 13
12 print(f'Predicción de ventas para el mes 13: {prediccion}')
```

2. INTRODUCCIÓN A R

- **R**

R es un lenguaje de programación y un entorno de software libre diseñado específicamente para el análisis estadístico y la visualización de datos.



2.1 Características Clave de R en el Análisis de Datos

Especialización en Análisis Estadístico

Potencia Estadística:

- R es reconocido por su robustez en análisis estadístico.
- Ofrece herramientas para pruebas estadísticas, modelado lineal y no lineal, análisis de series temporales, clasificación y clustering.

Funciones y Paquetes Estadísticos:

- Incluye una amplia colección de funciones integradas para análisis estadístico.
- Acceso a numerosos paquetes adicionales a través de CRAN (Comprehensive R Archive Network).

2.2 Visualización de Datos Avanzada

Capacidad de Visualización:

- R es conocido por su habilidad para generar gráficos de alta calidad.
- Desde gráficos simples (barras y líneas) hasta visualizaciones complejas (mapas de calor, gráficos de redes, gráficos interactivos).

Paquete ggplot2:

- Una de las herramientas más destacadas de R.
- Basado en la gramática de los gráficos, permite crear visualizaciones sofisticadas con facilidad.
- Ofrece una sintaxis coherente y flexible, convirtiéndose en el estándar de facto para la visualización en R.

2.2 Ejemplo De Código En R



```
1  # Cargar la biblioteca ggplot2, que se utiliza para crear gráficos en R
2  library(ggplot2)
3
4  # Crear un gráfico utilizando ggplot
5  # 'mpg' es un conjunto de datos que contiene información sobre el rendimiento de automóviles
6  # aes() define la estética del gráfico, asignando variables a los ejes y colores
7  ggplot(mpg, aes(x=displ, y=hwy, color=class)) +
8    # Agregar puntos al gráfico, donde cada punto representa un automóvil
9    geom_point() +
10   # Añadir etiquetas y título al gráfico
11   labs(title="Consumo de Combustible en Automóviles", # Título del gráfico
12         x="Desplazamiento del Motor (L)",           # Etiqueta del eje x
13         y="Millas por Galón en Carretera")          # Etiqueta del eje y
14
```

2.3 Amplia Colección de Paquetes y Librerías

- **Ecosistema Extenso:** CRAN alberga más de 18,000 paquetes, cubriendo áreas desde la biología computacional hasta la econometría, lo que hace que R sea extremadamente versátil y adecuado para una amplia variedad de disciplinas.
- **Paquetes Populares:** Además de ggplot2, otros paquetes populares incluyen dplyr (manipulación de datos), shiny (creación de aplicaciones web interactivas), caret (modelado predictivo), y tidyverse (colección de paquetes para la manipulación y visualización de datos en un estilo coherente y amigable)

2.4 Capacidades de Manipulación de Datos

- **dplyr y tidyr:** Estos paquetes forman parte del tidyverse y son fundamentales para la manipulación y transformación de datos en R. Con dplyr, los usuarios pueden filtrar, seleccionar, agrupar y resumir datos de manera eficiente, mientras que tidyr facilita la limpieza y estructuración de datos.

2.4 Ejemplo de Manipulación de Datos con dplyr



```
1  # Cargar la biblioteca dplyr, que se utiliza para la manipulación de datos
2  library(dplyr)
3
4  # Filtrar y resumir el conjunto de datos
5  datos_filtrados <- datos %>%
6    # Filtrar las filas donde la edad es mayor a 30
7    filter(edad > 30) %>%
8    # Agrupar los datos por género
9    group_by(género) %>%
10   # Calcular el promedio de ingreso para cada grupo
11   summarise(promedio_ingreso = mean(ingreso))
12
```

2.5 Interactividad y Creación de Aplicaciones

- **Shiny:** R permite la creación de aplicaciones web interactivas utilizando el paquete shiny. Estas aplicaciones permiten a los usuarios finales interactuar con los datos y visualizaciones de manera dinámica, lo que es útil para la presentación de análisis y la toma de decisiones basada en datos.
- **Integración con Markdown:** RMarkdown permite combinar texto, código y resultados en un solo documento, ideal para la generación de informes reproducibles que pueden ser exportados en múltiples formatos como HTML, PDF y Word.

2.5 Ejemplo de Aplicación Simple con shiny

```

1  # Cargar la biblioteca shiny, que se utiliza para construir aplicaciones web interactivas
2  library(shiny)
3
4  # Definición de la interfaz de usuario (UI)
5  ui <- fluidPage(
6    # Título de la aplicación
7    titlePanel("Aplicación de ejemplo"),
8
9    # Diseño de la interfaz con panel lateral y panel principal
10   sidebarLayout(
11     sidebarPanel(
12       # Control deslizante para seleccionar el número de observaciones
13       sliderInput("num", "Número de observaciones:",
14         min = 1, max = 100, value = 50)
15     ),
16     mainPanel(
17       # Área para mostrar el gráfico
18       plotOutput("hist")
19     )
20   )
21 )
22
23 # Definición del servidor
24 server <- function(input, output) {
25   output$hist <- renderPlot({
26     # Generar un histograma de números aleatorios de una distribución normal
27     hist(rnorm(input$num))
28   })
29 }
30
31 # Ejecutar la aplicación Shiny
32 shinyApp(ui = ui, server = server)

```

2.6 Entorno de Desarrollo Integrado (IDE):

- RStudio: RStudio es el entorno de desarrollo integrado más utilizado para R. Proporciona una interfaz amigable y potente que facilita la escritura de código, la manipulación de datos, la creación de gráficos, y la generación de informes. RStudio también se integra perfectamente con herramientas de control de versiones como Git.
- Funciones de RStudio: Entre sus características están la integración con proyectos, depuración de código, autocompletado, y la posibilidad de ejecutar fragmentos de código en una consola interactiva.

2.7 Soporte para Grandes Conjuntos de Datos y Computación Distribuida

- Aunque R originalmente fue diseñado para análisis en memoria, hay varios paquetes que permiten trabajar con grandes conjuntos de datos y realizar computación distribuida.
- Paquetes como data.table y sparklyr: Estos paquetes permiten el manejo eficiente de grandes conjuntos de datos y la integración con Apache Spark para computación distribuida.

2.8 Comunidad Activa y Recursos Abundantes:

- Comunidad Global: R cuenta con una comunidad global activa que contribuye constantemente con nuevos paquetes, tutoriales, y soluciones a problemas comunes. Esto garantiza que los usuarios siempre tengan acceso a las últimas innovaciones y mejoras en el lenguaje.
- Recursos Educativos: Hay una vasta cantidad de libros, cursos en línea, y documentación oficial disponibles, lo que facilita el aprendizaje y la adopción de R para nuevos usuarios.

2.9 Portabilidad y Compatibilidad::



- Multiplataforma: R es compatible con todos los principales sistemas operativos, incluidos **Windows, macOS y Linux**, lo que permite a los usuarios desarrollar y ejecutar código en cualquier entorno sin necesidad de cambios significativos.
- Compatibilidad con Otros Lenguajes: R puede integrarse con otros lenguajes de programación **como Python, C++, y SQL**, permitiendo a los usuarios aprovechar lo mejor de cada herramienta en proyectos complejos.

2.9 Ejemplo Práctico



```
1 # Carga de Datos: Utilizas read.csv() para cargar los datos del estudio en R.  
2 datos_presion <- read.csv('datos_presion_arterial.csv')  
3
```



```
1 # Exploración de Datos: Realizas un resumen estadístico de los datos.  
2 summary(datos_presion)  
3
```

2.9 Ejemplo Práctico



```
1 # Análisis Estadístico: Utilizas una prueba t para comparar la presión arterial promedio entre dos grupos.
2 t.test(datos_presion$presion ~ datos_presion$grupo)
3
```



```
1 # Visualización de Datos: Creas un gráfico de caja (boxplot) para visualizar la distribución de la presión arterial en diferentes grupos.
2 library(ggplot2)
3
4 ggplot(datos_presion, aes(x=grupo, y=presion)) +
5   geom_boxplot() +
6   labs(title="Distribución de la Presión Arterial por Grupo",
7        x="Grupo",
8        y="Presión Arterial (mm Hg)")
9
```



2.9 Ejemplo Práctico



```
1  # Generación de Informe: Utilizas RMarkdown para crear un informe reproducible con texto, código y gráficos.
2  ---
3  title: "Informe de Análisis de Presión Arterial"
4  author: "Nombre del Autor"
5  date: "`r Sys.Date()`"
6  output: html_document
7  ---
8
9  - **Introducción**
10 Este informe presenta un análisis de los datos de presión arterial...
11
12 ```{r}
13 # Cargar y analizar los datos
14 library(ggplot2)
15 datos_presion <- read.csv('datos_presion_arterial.csv')
16 summary(datos_presion)
```

2.9 Ejemplo Práctico



```
1  # Visualización de la Presión Arterial
2
3  ggplot(datos_presion, aes(x=grupo, y=presion)) +
4    geom_boxplot() +
5    labs(title="Distribución de la Presión Arterial por Grupo",
6         x="Grupo",
7         y="Presión Arterial (mm Hg)")
8
```


2.9 Conclusión

R es un lenguaje de programación excepcionalmente poderoso y flexible, ideal para cualquier tarea relacionada con el análisis estadístico y la visualización de datos. Su extensa colección de paquetes, su capacidad para crear gráficos de alta calidad, y su enfoque especializado en estadísticas lo convierten en una herramienta indispensable para estadísticos, científicos de datos, y analistas en una variedad de campos. Con R, los usuarios pueden realizar análisis complejos, generar visualizaciones informativas y comunicar sus hallazgos de manera efectiva, todo dentro de un entorno unificado y reproducible.

▶ TALENTO
**¡Prepárate para desafiar los límites
de bootcamp de análisis de datos!**