Statistics and Data Analysis Using R

Organised by Unilorin R-Users

Dr Isaac Ajao

2025-03-22

What to learn in this training

1. Introduction to R for Statistics and Data Analysis

- Overview of R and its importance in statistics & data analysis
- Installing R and RStudio
- Basic R syntax (variables, data types, functions)
- Loading essential libraries (`tidyverse`, `ggplot2`, `dplyr`)

2. Core Statistical Techniques Using R

- **Descriptive Statistics**: Mean, median, mode, variance, standard deviation (`summary()`, `sd()`, `IQR()`)
- **Data Visualization**: Histogram, boxplot, scatterplot (`ggplot2`)
- **Hypothesis Testing**: t-tests, ANOVA (`t.test()`, `aov()`)
- **Regression Analysis**: Simple & multiple linear regression (`lm()`, `summary()`)

3. Data Analysis Workflow

- **Importing Data**: CSV, Excel, database connections (`read.csv()`, `readxl`)
- **Data Cleaning**: Handling missing values (`na.omit()`, `tidyverse` functions)
- **Data Transformation**: Filtering, selecting, mutating columns (`dplyr::filter()`, `mutate()`)
- **Exploratory Data Analysis (EDA)**: Summarizing and visualizing key patterns

4. Real-World Use Case: Data-Driven Decision Making

- A case study (e.g., analyzing customer satisfaction, sales trends, environmental data)
- Walk through the full process: data import $\rightarrow$ cleaning $\rightarrow$ analysis $\rightarrow$ visualization $\rightarrow$ interpretation
- Discussion on insights and how to communicate findings effectively

1. Introduction to R for Statistics and Data Analysis

Overview of **R**

R is a powerful open-source programming language designed primarily for statistical computing and data analysis. It provides a rich ecosystem of packages and built-in functions for handling, analyzing, and visualizing data. R is widely used in academia, research, and industries such as finance, healthcare, and machine learning.

Importance of R in Statistics & Data Analysis

Statistical Modeling – **R** has robust libraries for regression, hypothesis testing, ANOVA, time series analysis, and Bayesian modeling.

Data Manipulation – Packages like `dplyr` and `tidyverse` make it easy to clean, transform, and manipulate datasets.

Data Visualization – `ggplot2` is a leading tool for creating high-quality visualizations.

Machine Learning – **R** supports ML techniques via `caret`, `randomForest`, and `xgboost`.

Reproducible Research – Quarto and RMarkdown allow users to document analysis in a report-friendly format.

Big Data & Integration – **R** can handle large datasets and integrates well with databases, Python, and cloud platforms.

Downloading and Installing **R**

Download R here **https://cloud.r-project.org/**

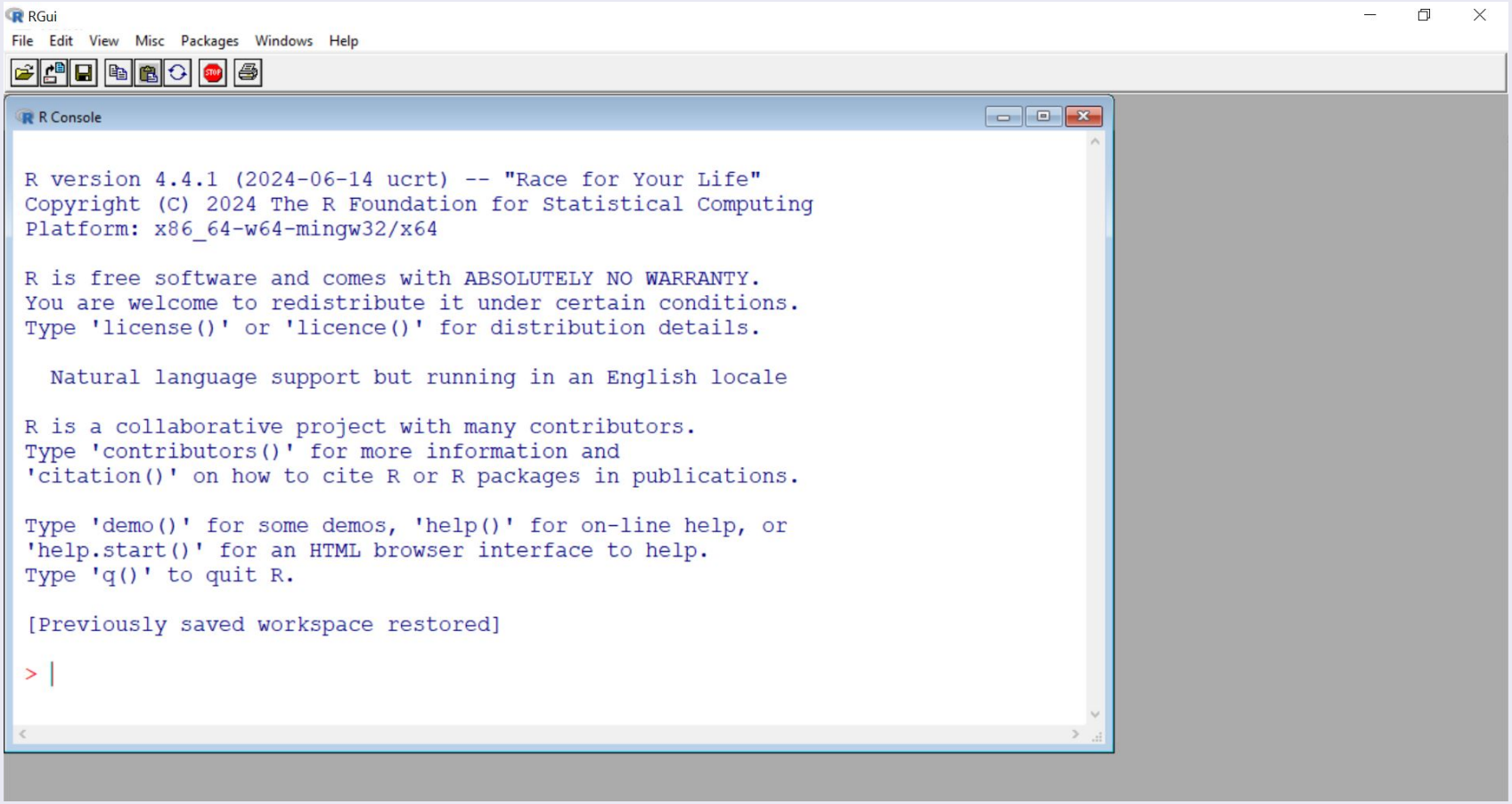

Figure 1: R Console

R Interface

The first interface you see after installing and opening **R** is called the **R Console**.

The **R Console** is a command-line interface where you can type and execute R commands interactively. It allows you to run scripts, perform calculations, and see immediate outputs.

Downloading and Installing RStudio

Download RStudio here

https://posit.co/download/rstudio-desktop/

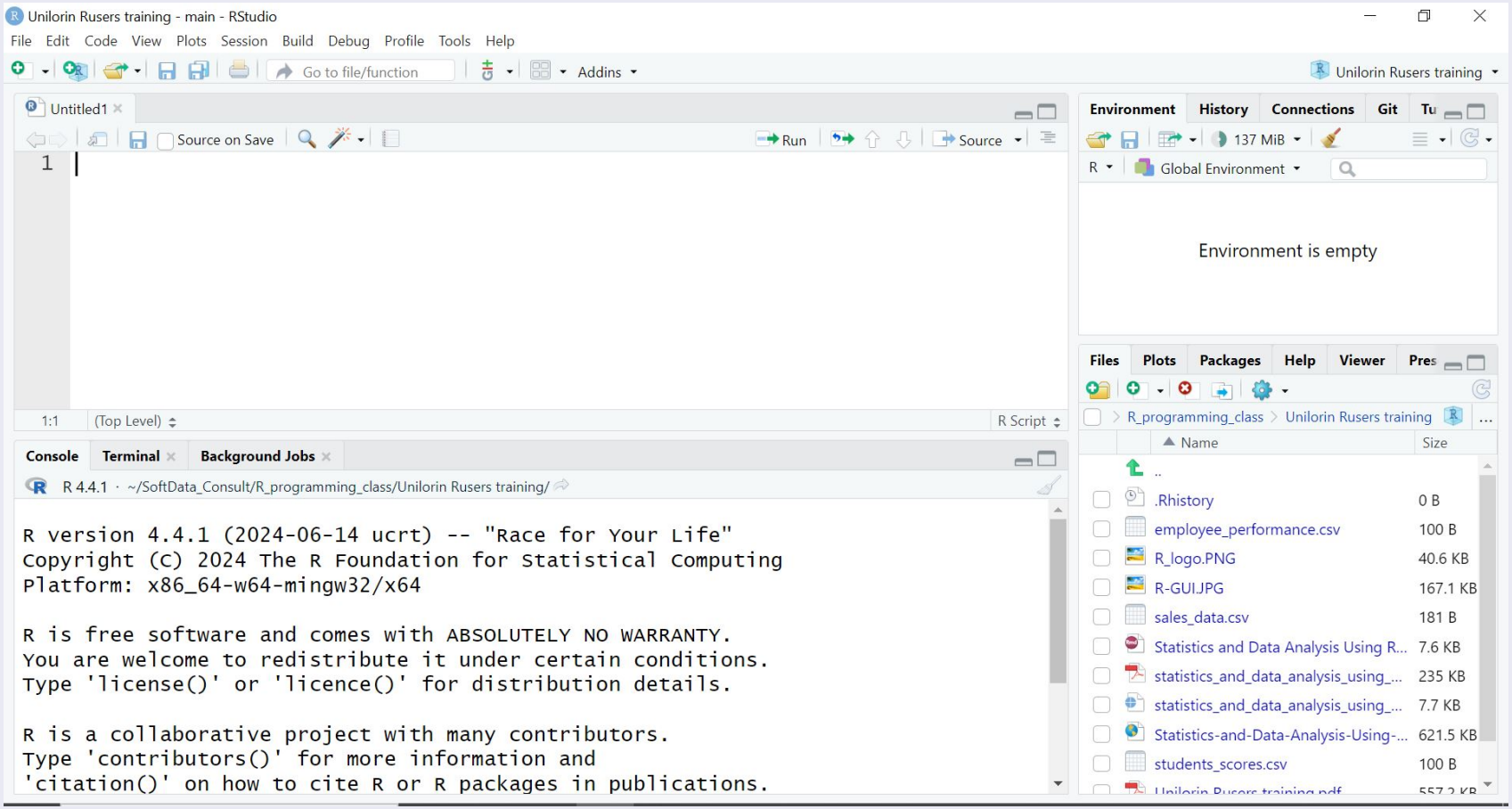

Figure 2: R Studio IDE

RStudio Interface

In **RStudio**, the first interface you see after launching it is the **RStudio IDE (Integrated Development Environment)**. It consists of four main panes:

1. **Source Editor (Top-Left)** – For writing and editing R scripts (.R files).
2. **Console (Bottom-Left)** – Where R commands are executed interactively.
3. **Environment/History (Top-Right)** – Displays variables, data frames, and command history.
4. **Files/Plots/Packages/Help (Bottom-Right)** – Manages files, plots, installed packages, and documentation.

The **Console** is the core execution environment, but RStudio provides additional tools to make coding easier.

R Script: Basic operations

```r
# Install and load essential libraries
install.packages("tidyverse")
library(tidyverse)

# Basic operations
x <- c(10, 20, 30, 40, 50)
mean(x)   # Compute mean
sd(x)     # Compute standard deviation
summary(x)  # Get a statistical summary
```

Dataset: `students_scores.csv` (Contains student names, test scores, and study hours)

Name	Score	Study_Hours
Ade	85	10
Chuks	78	8
Bayo	90	12
Mary	70	6
Afusat	88	11

R Script:

```r
# Load the dataset
students <- read.csv("students_scores.csv")

# Descriptive statistics
summary(students)

# Histogram of scores
ggplot(students, aes(x = Score)) + geom_histogram(binwidth
= 5, fill = "blue", color = "black")

# Hypothesis test (t-test)
t.test(students$Score, mu = 75)  # Checking if the average

# Linear regression: Study Hours vs Score
model <- lm(Score ~ Study_Hours, data = students)
summary(model)
```

3. Data Analysis Workflow

Dataset: `sales_data.csv` (Contains Date, Sales, and Product Category)

Date	Sales	Category
2024-01-01	500	Electronics
2024-01-02	700	Clothing
2024-01-03	800	Electronics
2024-01-04	400	Clothing
2024-01-05	650	Electronics

R Script:

```r
# Load dataset
sales <- read.csv("sales_data.csv")

# Convert Date column to Date format
sales$Date <- as.Date(sales$Date, format="%Y-%m-%d")

summary(sales) # Summary statistics

# Filter sales for Electronics only
electronics_sales <- filter(sales,
Category == "Electronics")

# Aggregate total sales by category
total_sales <- sales %>% group_by(Category) %>%
summarise(Total = sum(Sales))
print(total_sales)

# Plot time series sales trend
```

4. Real-World Use Case: Data-Driven Decision Making

Case Study: Predicting Employee Performance Based on Work Hours

Dataset: `employee_performance.csv`

Employee	Work_Hours	Performance_Score
A	35	78
B	40	85
C	45	90
D	50	92
E	38	80

R Script:

```r
# Load dataset
performance <- read.csv("employee_performance.csv")

# Scatter plot
ggplot(performance, aes(x = Work_Hours, y =
Performance_Score)) + geom_point() +
geom_smooth(method="lm")

# Linear regression model
perf_model <- lm(Performance_Score ~ Work_Hours,
data = performance)
summary(perf_model)

# Predict performance for a new employee working 42 hours
new_data <- data.frame(Work_Hours = 42)
predict(perf_model, new_data)
```

How to Load These Files in R

Once you've saved the datasets, you can load them using:

```
# students score data
students <- read.csv("students_scores.csv")

# sales data
sales <- read.csv("sales_data.csv")

# employee data
performance <- read.csv("employee_performance.csv")
```

Next Up: Hands-On Practical Session

Now it's time to apply what we've learned!

Figure 3: R logo

Figure 4: RStudio Logo

- We will explore real-world data and practice key concepts.
- Follow along with the guided exercises and try running the code yourself.
- Feel free to ask questions as we go!

Let's dive into the practical session and bring our knowledge to life!