Lecture 9: Working with Data Frames, Lists, and Formulas in R

Dr. Logan Kelly

2024-09-04

Overview

- In this lecture, we'll cover:
  - How to **load and save data** in R using **data frames**.
  - Working with **lists** to manage complex data.
  - How to use **formulas** for modeling and statistical analysis.

1. Loading and Saving Data

- **What is a data frame?**
  - A **data frame** is a table-like structure where each column can hold different types of data (numeric, character, etc.), making it ideal for storing datasets.
- **Loading data into a data frame**:
  - You can load external data, such as a CSV file, into a data frame using the `read.csv()` function.
- **Example: Loading a CSV file**:

```
data <- read.csv("data.csv")
```

- **Saving data from a data frame**:
  - You can save a data frame to a CSV file using the `write.csv()` function.
- **Example: Saving a data frame to a CSV file**:

```
write.csv(data, "output.csv")
```

- **Viewing the structure of a data frame**:
  - The `str()` function shows the structure of the data frame, including the types of variables and the first few rows of data.
- **Example: Viewing the structure of a data frame**:

```
str(data)
```

```
function (..., list = character(), package = NULL, lib.loc = NULL, verbose = getOption("verbo
    envir = .GlobalEnv, overwrite = TRUE)
```

2. Working with Lists

- **What is a list in R?**
  - A **list** is a flexible data structure that can store objects of different types, including vectors, data frames, and other lists.
- **How to create a list**:
  - Use the `list()` function to create a list that holds various elements.
- **Example: Creating a list with different data types**:

```
my_list <- list(
  name = "Alice",
  age = 25,
  salary = 50000,
  skills = c("R", "Python", "SQL")
)
```

- **Accessing elements in a list**:
  - Use double square brackets `[[ ]]` or the `$` operator to access elements in a list.
- **Example: Accessing the `name` element**:

```
my_list$name
```

```
[1] "Alice"
```

- **Example: Accessing the `skills` vector**:

```
my_list$skills
```

```
[1] "R"      "Python" "SQL"
```

- **Modifying elements in a list**:
  - You can modify existing elements by assigning new values.
- **Example: Changing the `salary` element**:

```
my_list$salary <- 60000
```

3. Using Formulas in R

- **What is a formula?**
  - A **formula** in R is used to express relationships between variables, especially in statistical modeling.
  - The general syntax for a formula is `y ~ x`, where `y` is the dependent variable and `x` is the independent variable.
- **Example: Creating a simple formula**:

```
# A formula for predicting mpg using cyl and disp
formula <- mpg ~ cyl + disp
```

- **Formulas in regression analysis**:
  - Formulas are widely used in **regression analysis** to specify models.
- **Example: Using a formula in a linear regression**:

```
model <- lm(mpg ~ cyl + disp, data = mtcars)
summary(model)  # View the summary of the regression model
```

```

Call:
lm(formula = mpg ~ cyl + disp, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.4213 -2.1722 -0.6362  1.1899  7.0516 
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 34.66099    2.54700  13.609 4.02e-14 ***
cyl         -1.58728    0.71184  -2.230   0.0337 *
disp        -0.02058    0.01026  -2.007   0.0542 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.055 on 29 degrees of freedom
Multiple R-squared:  0.7596,	Adjusted R-squared:  0.743
F-statistic: 45.81 on 2 and 29 DF,  p-value: 1.058e-09
```

- **Explanation**:
  - The formula `mpg ~ cyl + disp` tells R to model `mpg` (miles per gallon) as a function of `cyl` (cylinders) and `disp` (displacement).
  - The `lm()` function fits a linear model, and `summary()` provides the model's details.

4. Combining Data Frames, Lists, and Formulas

- **Lists of data frames**:
  - You can store multiple data frames in a list and use formulas to perform statistical analysis on each one.
- **Example: Creating a list of data frames and applying a formula**:

```r
df1 <- data.frame(name = c("Alice", "Bob"), age = c(25, 30))
df2 <- data.frame(name = c("Charlie", "David"), age = c(35, 40))
data_list <- list(df1, df2)

# Applying a simple operation to one of the data frames
mean_age_df1 <- mean(data_list[[1]]$age)
```

- **Using formulas with data in lists**:
  - You can apply formulas to perform calculations or modeling on data stored in lists.
- **Example: Applying a formula to a data frame in a list**:

```r
model_df1 <- lm(age ~ name, data = data_list[[1]])
```

Key Takeaways

- **Data frames** allow you to load, store, and manipulate tabular data in R.
- **Lists** are flexible and can store a variety of data types, making them useful for managing complex datasets.
- **Formulas** are essential for statistical analysis and are commonly used in models like regression.

Looking Forward

- In the next lecture, we'll dive into **programming in R**, focusing on writing scripts, using loops and conditional statements, and writing custom functions to solve more advanced problems.