Lecture 7: Working with Lists and Factors in R

Dr. Logan Kelly

2024-09-04

Overview

- In this lecture, we'll cover:
  - Introduction to **lists** and their flexibility in R.
  - How to create, access, and modify **lists**.
  - Working with **factors**, a key data structure for categorical data.
  - Practical examples of using lists and factors in business data analysis.

1. Introduction to Lists

- A **list** in R is a flexible data structure that can contain elements of **different types**, including vectors, matrices, data frames, and even other lists.
- Lists are highly versatile, making them useful for managing complex collections of data.

Creating a List

You can create a list using the `list()` function, which combines various objects into a single list.

Example: Creating a Sales List

```
# Creating a list with different types of data
sales_list <- list(
  Products = c("A", "B", "C"),
  Sales_Q1 = c(120, 150, 90),
  Sales_Matrix = matrix(c(120, 150, 90, 100, 130, 170), nrow = 3, ncol = 2)
)
sales_list
```

```
$Products
[1] "A" "B" "C"

$Sales_Q1
[1] 120 150  90

$Sales_Matrix
     [,1] [,2]
[1,]  120  100
[2,]  150  130
[3,]   90  170
```

- **Explanation**: This list contains a vector of product names, a vector of sales for the first quarter, and a matrix of sales data across multiple periods.

Accessing Elements in a List

You can access elements in a list using either: - The `$` operator for named elements. - Double square brackets `[[ ]]` for positional access.

Example: Accessing List Elements by Name

```
# Accessing the Products vector in the list
sales_list$Products
```

```
[1] "A" "B" "C"
```

Example: Accessing List Elements by Position

```
# Accessing the Sales_Q1 vector (second element) in the list
sales_list[[2]]
```

```
[1] 120 150  90
```

- **Explanation**: This code accesses the second element in the list, which is the `Sales_Q1` vector.

Modifying Elements in a List

You can modify or add elements to a list by assigning new values to a specific position or name.

Example: Adding a New Element to the List

```
# Adding a new element for Sales in Q2
sales_list$Sales_Q2 <- c(170, 200, 140)
sales_list
```

```
$Products
[1] "A" "B" "C"

$Sales_Q1
[1] 120 150  90

$Sales_Matrix
     [,1] [,2]
[1,]  120  100
[2,]  150  130
[3,]   90  170

$Sales_Q2
[1] 170 200 140
```

- **Explanation**: This adds a new vector, `Sales_Q2`, to the `sales_list`, which now contains sales data for both Q1 and Q2.

Removing Elements from a List

You can remove an element from a list by setting it to `NULL`.

Example: Removing an Element from a List

```
# Removing the Sales_Matrix from the list
sales_list$Sales_Matrix <- NULL
sales_list
```

```
$Products
[1] "A" "B" "C"

$Sales_Q1
[1] 120 150  90

$Sales_Q2
[1] 170 200 140
```

- **Explanation**: This code removes the `Sales_Matrix` element from the list.

2. Introduction to Factors

- **Factors** are used in R to store **categorical data**, which is data that falls into a limited number of categories (e.g., "High", "Medium", "Low").
- Factors are especially useful in **statistical analysis** and **modeling**, ensuring that categorical variables are properly treated.

Creating a Factor

You can create a factor using the `factor()` function.

Example: Creating a Factor for Customer Satisfaction

```
# Creating a factor for customer satisfaction levels
satisfaction_levels <- factor(c("High", "Medium", "Low", "Medium", "High"))
satisfaction_levels
```

```
[1] High   Medium Low    Medium High
Levels: High Low Medium
```

- **Explanation**: This factor contains customer satisfaction levels (e.g., "High", "Medium", "Low") and treats them as distinct categories.

Working with Levels in a Factor

Factors have **levels**, which represent the different categories within the data. You can view or modify these levels.

Example: Viewing the Levels of a Factor

```
# Viewing the levels of the satisfaction factor
levels(satisfaction_levels)
```

```
[1] "High"   "Low"    "Medium"
```

- **Explanation**: This returns the levels ("High", "Medium", "Low") of the `satisfaction_levels` factor.

Example: Changing the Order of Levels

```
# Reordering the levels of the satisfaction factor
satisfaction_levels <- factor(satisfaction_levels, levels = c("Low", "Medium", "High"))
satisfaction_levels
```

```
[1] High   Medium Low    Medium High
Levels: Low Medium High
```

- **Explanation**: This reorders the levels of the `satisfaction_levels` factor to reflect a logical progression from "Low" to "High."

Working with Ordered Factors

If the categories in a factor have a meaningful **order**, such as "Poor", "Fair", "Good", "Excellent", you can create an **ordered factor**.

Example: Creating an Ordered Factor for Customer Satisfaction

```
# Creating an ordered factor
ordered_satisfaction <- factor(c("Good", "Excellent", "Fair", "Good", "Poor"),
                               levels = c("Poor", "Fair", "Good", "Excellent"),
                               ordered = TRUE)
ordered_satisfaction
```

```
[1] Good      Excellent Fair      Good      Poor
Levels: Poor < Fair < Good < Excellent
```

- **Explanation**: This ordered factor treats the satisfaction levels as having a ranked order, from "Poor" to "Excellent."

3. Practical Example: Using Lists and Factors in Business Analytics

Example: Combining Product Data in a List

You can use lists to store and organize different types of data related to business operations.

```r
# Creating a list that combines product names, sales, and satisfaction ratings
business_data <- list(
  Products = c("A", "B", "C"),
  Sales = c(150, 200, 130),
  Satisfaction = factor(c("High", "Medium", "Low"))
)
business_data
```

```
$Products
[1] "A" "B" "C"

$Sales
[1] 150 200 130

$Satisfaction
[1] High   Medium Low
Levels: High Low Medium
```

- **Explanation**: This list contains product names, sales data, and customer satisfaction levels for each product.

Example: Analyzing Satisfaction Ratings Using Factors

```r
# Summarizing satisfaction levels
summary(business_data$Satisfaction)
```

```
  High    Low Medium
     1      1      1
```

- **Explanation**: This summary provides a count of the different satisfaction levels ("High", "Medium", "Low") in the `business_data` list.

Key Takeaways

- **Lists** are flexible data structures that can hold different types of elements, making them ideal for handling complex data.
- **Factors** are used to store and manage **categorical data**, ensuring proper handling of non-numeric variables in statistical models.
- You now know how to create, access, modify, and use lists and factors in R for business data analysis.

Looking Forward

- In the next lecture, we'll explore how to **import, clean, and transform** business datasets in R, a key skill for data preparation before analysis.