LabExercises_Week01

Alison Lawyer

2024-09-12

keep this chunk in all your RMarkdown scripts

```r
knitr::opts_chunk$set(echo = TRUE)
knitr::opts_chunk$set(tidy.opts = list(width.cutoff = 60), tidy = TRUE)
```

```r
# Lab Objectives
# In today's lab, we will:
# - Explore different types of data structures in R.
# - Import a dataset and perform basic data inspection.
# - Manipulate data using functions from the `tidyverse` package.
# - Reshape data using `pivot_longer()` and `pivot_wider()`.
# - Categorize continuous data into factors using the `cut()` function.
# - Handle missing data (NA values).

# Setup: Loading Required Packages
# Install and load the required packages
# install.packages("tidyverse")
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
# Step 1: Importing a Dataset
# We will use a built-in dataset `iris` in this lab, which contains data on the characteristics of iris

# Load the iris dataset
data("iris")

# View the first few rows of the dataset
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
# Inspect the dataset:
# - What are the dimensions of the dataset?
# - How are the variables structured?

# Dimensions of the dataset
dim(iris)
```

```
## [1] 150   5
```

```
# Summary statistics of the dataset
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
##                 
```

```
# Data structure
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
# Step 2: Basic Data Manipulation
# The `tidyverse` offers powerful functions to manipulate and clean your data. Let's explore `filter()`

# Select specific columns (e.g., only Sepal and Species columns)
iris_selected <- select(iris, Sepal.Length, Sepal.Width, Species)
head(iris_selected)
```

```
##   Sepal.Length Sepal.Width Species
## 1          5.1         3.5  setosa
## 2          4.9         3.0  setosa
## 3          4.7         3.2  setosa
## 4          4.6         3.1  setosa
## 5          5.0         3.6  setosa
## 6          5.4         3.9  setosa
```

```r
# Filter the dataset to only include flowers with Sepal.Length greater than 5
iris_filtered <- filter(iris, Sepal.Length > 5)
head(iris_filtered)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          5.4         3.9          1.7         0.4  setosa
## 3          5.4         3.7          1.5         0.2  setosa
## 4          5.8         4.0          1.2         0.2  setosa
## 5          5.7         4.4          1.5         0.4  setosa
## 6          5.4         3.9          1.3         0.4  setosa
```

```r
# Arrange the dataset by Sepal.Length in descending order
iris_arranged <- arrange(iris, desc(Sepal.Length))
head(iris_arranged)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 1          7.9         3.8          6.4         2.0 virginica
## 2          7.7         3.8          6.7         2.2 virginica
## 3          7.7         2.6          6.9         2.3 virginica
## 4          7.7         2.8          6.7         2.0 virginica
## 5          7.7         3.0          6.1         2.3 virginica
## 6          7.6         3.0          6.6         2.1 virginica
```

```r
# Create new variables using `mutate()`. Let's calculate the ratio of Sepal.Length to Sepal.Width
iris <- mutate(iris, Sepal.Ratio = Sepal.Length / Sepal.Width)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species Sepal.Ratio
## 1          5.1         3.5          1.4         0.2  setosa    1.457143
## 2          4.9         3.0          1.4         0.2  setosa    1.633333
## 3          4.7         3.2          1.3         0.2  setosa    1.468750
## 4          4.6         3.1          1.5         0.2  setosa    1.483871
## 5          5.0         3.6          1.4         0.2  setosa    1.388889
## 6          5.4         3.9          1.7         0.4  setosa    1.384615
```

```r
# Step 3: Categorizing Data with `cut()`
# We can categorize continuous variables using `cut()`. Let’s create categories for the `Sepal.Length`

# Categorize Sepal.Length into bins: "Short", "Medium", and "Long"
iris <- iris %>% mutate(Sepal.Length.Category =
                        cut(Sepal.Length,
                            breaks = c(4, 5.5, 6.5, 8),
                            labels = c("Short", "Medium", "Long")))
```

```r
# Check the distribution of the new categories
table(iris$Sepal.Length.Category) # count indiv rows in each category
```

```
##
##  Short Medium   Long
##     59     61     30
```

```r
# Step 4: Reshaping Data with `pivot_longer()` and `pivot_wider()`
# Reshaping data is an important concept for manipulating datasets for analysis. We will reshape the ir

# Pivot data from wide to long format
iris_long <- iris %>% mutate(sample = row_number()) %>%
  pivot_longer(cols = Sepal.Length:Petal.Width,
                                   names_to = "Measurement",
                                   values_to = "Value")
head(iris_long)
```

```
## # A tibble: 6 x 6
##   Species Sepal.Ratio Sepal.Length.Category sample Measurement  Value
##   <fct>         <dbl> <fct>                  <int> <chr>        <dbl>
## 1 setosa         1.46 Short                      1 Sepal.Length   5.1
## 2 setosa         1.46 Short                      1 Sepal.Width    3.5
## 3 setosa         1.46 Short                      1 Petal.Length   1.4
## 4 setosa         1.46 Short                      1 Petal.Width    0.2
## 5 setosa         1.63 Short                      2 Sepal.Length   4.9
## 6 setosa         1.63 Short                      2 Sepal.Width    3
```

```r
# Now, reshape it back to wide format using `pivot_wider()`
iris_wide <- iris_long %>% pivot_wider(names_from = Measurement,
                                       values_from = Value)
head(iris_wide)
```

```
## # A tibble: 6 x 8
##   Species Sepal.Ratio Sepal.Length.Category sample Sepal.Length Sepal.Width
##   <fct>         <dbl> <fct>                  <int>        <dbl>       <dbl>
## 1 setosa         1.46 Short                      1          5.1         3.5
## 2 setosa         1.63 Short                      2          4.9         3
## 3 setosa         1.47 Short                      3          4.7         3.2
## 4 setosa         1.48 Short                      4          4.6         3.1
## 5 setosa         1.39 Short                      5          5           3.6
## 6 setosa         1.38 Short                      6          5.4         3.9
## # i 2 more variables: Petal.Length <dbl>, Petal.Width <dbl>
```

```r
# Step 5: Handling Missing Data
# Although the `iris` dataset doesn't have missing data, we will simulate a scenario where there are so

# Create a copy of the iris dataset and introduce NA values
iris_with_na <- iris
set.seed(123) # For reproducibility (but how?)
iris_with_na[sample(1:nrow(iris_with_na), 10), "Sepal.Length"] <- NA
head(iris_with_na)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species Sepal.Ratio
## 1          5.1         3.5          1.4         0.2  setosa    1.457143
## 2          4.9         3.0          1.4         0.2  setosa    1.633333
## 3          4.7         3.2          1.3         0.2  setosa    1.468750
## 4          4.6         3.1          1.5         0.2  setosa    1.483871
## 5          5.0         3.6          1.4         0.2  setosa    1.388889
## 6          5.4         3.9          1.7         0.4  setosa    1.384615
##   Sepal.Length.Category
## 1                 Short
## 2                 Short
## 3                 Short
## 4                 Short
## 5                 Short
## 6                 Short
```

```
# Identify rows with `NA` values
iris_with_na %>% filter(is.na(Sepal.Length))
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width    Species Sepal.Ratio
## 1            NA         3.0          1.1         0.1     setosa    1.433333
## 2            NA         3.2          1.3         0.2     setosa    1.375000
## 3            NA         3.3          1.4         0.2     setosa    1.515152
## 4            NA         2.5          4.0         1.3 versicolor    2.200000
## 5            NA         2.6          4.4         1.2 versicolor    2.115385
## 6            NA         3.0          4.6         1.4 versicolor    2.033333
## 7            NA         3.8          6.7         2.2  virginica    2.026316
## 8            NA         2.7          5.1         1.9  virginica    2.148148
## 9            NA         3.0          5.2         2.0  virginica    2.166667
## 10           NA         3.0          5.1         1.8  virginica    1.966667
##    Sepal.Length.Category
## 1                  Short
## 2                  Short
## 3                  Short
## 4                  Short
## 5                  Short
## 6                 Medium
## 7                   Long
## 8                 Medium
## 9                 Medium
## 10                Medium
```

```
# Remove rows with `NA` values
iris_no_na <- iris_with_na %>% drop_na(Sepal.Length)
head(iris_no_na)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species Sepal.Ratio
## 1          5.1         3.5          1.4         0.2  setosa    1.457143
## 2          4.9         3.0          1.4         0.2  setosa    1.633333
## 3          4.7         3.2          1.3         0.2  setosa    1.468750
## 4          4.6         3.1          1.5         0.2  setosa    1.483871
## 5          5.0         3.6          1.4         0.2  setosa    1.388889
## 6          5.4         3.9          1.7         0.4  setosa    1.384615
##   Sepal.Length.Category
## 1                  Short
## 2                  Short
## 3                  Short
## 4                  Short
## 5                  Short
## 6                  Short
```

```
# Fill `NA` values with the mean of the Sepal.Length variable
iris_filled <- iris_with_na %>% mutate(Sepal.Length = ifelse(is.na(Sepal.Length),
                                                              mean(Sepal.Length, na.rm = TRUE),
                                                              Sepal.Length))
head(iris_filled)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species Sepal.Ratio
## 1          5.1         3.5          1.4         0.2  setosa    1.457143
## 2          4.9         3.0          1.4         0.2  setosa    1.633333
## 3          4.7         3.2          1.3         0.2  setosa    1.468750
## 4          4.6         3.1          1.5         0.2  setosa    1.483871
## 5          5.0         3.6          1.4         0.2  setosa    1.388889
## 6          5.4         3.9          1.7         0.4  setosa    1.384615
##   Sepal.Length.Category
## 1                  Short
## 2                  Short
## 3                  Short
## 4                  Short
## 5                  Short
## 6                  Short
```

```
# Step 6: Save the Cleaned Dataset
# It's always a good idea to save your manipulated dataset.

# Save the modified dataset to your working directory
write.csv(iris_filled, "iris_cleaned.csv")

# Summary
# In this lab, we covered the following concepts:
# - Basic data exploration and inspection.
# - Selecting, filtering, and arranging data with `tidyverse` functions.
# - Creating new variables with `mutate()`.
# - Categorizing continuous variables into factors using `cut()`.
# - Reshaping data using `pivot_longer()` and `pivot_wider()`.
# - Handling missing values using `drop_na()` and filling missing values with `mutate()`.
# These are essential skills in data preparation and manipulation for ecology and evolutionary biology
```