Week-3: Code-along

NMI2207: Computational Media Literacy

2023-08-24

Welcome! Go through the steps described below, carefully. It is totally fine to get stuck - **ASK FOR HELP!** reach out to your friends, TAs, or the discussion forum on Canvas.

Here is what you have to do,

1. **Download** `Code-along.Rmd` and `cat_lovers.csv` files from Canvas and move it to the folder "Week-3" (see instructions for creating folder in Section I below)
2. **Open** the video lectures and start listening to them
3. **Every time** you come across a code chunk (inside shaded blocks) in the lecture video, **Pause the video**
4. **Edit** the `Code-along.Rmd` file with the codes explained in the lecture videos within appropriate R chunk(code-block/shaded area (environment enclosed within ```)
5. **Comments** inside the R chunk/code-block/shaded area indicates which command explained in the lecture should be typed in there
6. **Set** `eval=TRUE` to generate the output and verify it to the one shown in the lecture videos
7. **Knit** the file upon completion and submit the pdf document on Canvas **before** coming to the tutorial session

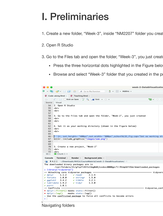
I. Preliminaries

1. Create a new folder, "Week-3", inside "NMI2207" folder you created last week

2. Open R Studio

3. Go to the File tab and open the folder, "Week-3", you just created

- Press the three horizontal dots highlighted in the Figure below
- Browse and select "Week-3" folder that you created in the previous step, inside "NMI2207" folder



Navigating folders

4. Set it as your working directory (shown in the Figure below)

Set as working directory

5. Now, create a new project and name it "Week-3"

Figure: Creating a new project

6. Download the `Code-along.Rmd` file from Canvas and move it to t

II. Code to edit and execute using the Code-along.Rmd file

Loading packages

```
# load package tidyverse
```

Assigning values to variables

```
# Example x,y execute this example
x <- 'A'
x
```

```
# Complete the code for Example b and execute it
```

```
# Complete the code for Example c and execute it
```

```
# Complete the code for Example d and execute it
```

```
# Complete the code for Example e and execute it
```

```
# Complete the code for Example f and execute it
```

Checking the type of variables

```
# Example x,y execute this example
x <- 'A'
typeof(x)
```

```
# Complete the code for Example b and execute it
```

```
# Complete the code for Example c and execute it
```

```
# Complete the code for Example d and execute it
```

```
# Complete the code for Example e and execute it
```

```
# Complete the code for Example f and execute it
```

Need for data types

```
# Import the cat-lovers data from the csv file you downloaded from canvas
```

```
# Compute the mean of the number of cats: execute this command
mean(cat_lovers$number_of_cats)
```

```
# Get more information about the mean() command using ? operator
```

```
# Convert the variable number_of_cats using as.integer()
```

```
# Display the elements of the column number_of_cats
```

```
# Display the elements of the column number_of_cats after converting it using as.numeric()
```

Create an empty vector

```
# Empty vector
# Type of the empty vector
typeof(x)
```

Create vectors of type logical

```
# Method 1
x<-vector("logical",length=5)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```
# Method 2
x<-logical(5)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

```
# Method 3
x<-c(TRUE,FALSE,TRUE,FALSE,TRUE)
# Display the contents of x
print(x)
# Display the type of x
print(typeof(x))
```

Create vectors of type character

```
# Method 1
```

```
# Display the contents of x
```

```
# Display the type of x
print(typeof(x))
```

```
# Method 2
```

```
# Display the contents of x
print(x)
# Display the type of x
```

```
# Method 3
```

```
# Display the contents of x
```

```
# Display the type of x
```

Create vectors of type integer

```
# Method 1
```

```
# Display the contents of x
```

```
# Display the type of x
print(typeof(x))
```

```
# Method 2
```

```
# Display the contents of x
print(x)
# Display the type of x
```

```
# Method 3
```

```
# Display the contents of x
```

```
# Display the type of x
```

```
# Method 4
```

```
# Display the contents of x
```

```
# Display the type of x
```

```
# Method 5
```

```
# Display the contents of x
```

```
# Display the type of x
```

Create vectors of type double

```
# Method 1
```

```
# Display the contents of x
```

```
# Display the type of x
```

```
# Method 2
```

```
# Display the contents of x
```

```
# Display the type of x
```

```
# Method 3
```

```
# Display the contents of x
```

```
# Display the type of x
```

Implicit coercion

Example 1

```
# Create a vector
```

```
# Check the type of x
```

```
# Add a character to the vector
```

```
# Check the type of x
```

Example 2

```
# Create a vector
```

```
# Check the type of x
```

```
# Add a number to the vector
```

```
# Check the type of x
```

Example 3

```
# Create a vector
```

```
# Check the type of x
```

```
# Add a logical value to the vector
```

```
# Check the type of x
```

Example 4

```
# Create a vector
```

```
# Check the type of x
```

```
# Add a number to the vector
```

```
# Check the type of x
```

Explicit coercion

Example 1

```
# Create a vector
```

```
# Check the type of x
```

```
# Convert the vector to type character
```

```
# Check the type of x
```

Example 2

```
# Create a vector
```

```
# Check the type of x
```

```
# Convert the vector to type double
```

```
# Check the type of x
```

Accessing elements of the vector

```
# Create a vector
x <- c(1,10,5,8,1,1,5)
```

```
# Access one element with index 3
```

```
# Access elements with consecutive indices, 2 to 4: 2,3,4
```

```
# Access elements with non-consecutive indices, 1,3,5
```

```
# Access elements using logical vector
x[c(TRUE,FALSE,FALSE,TRUE,FALSE,FALSE,TRUE)]
```

```
# Access elements using the conditional operator <
```

Examining vectors

```
# Display the length of the vector
print(length(x))
# Display the type of the vector
print(typeof(x))
# Display the structure of the vector
print(str(x))
```

Lists

```
# Initialize a named list
my_pie = list(type="key lime", diameter=7, is.vegetarian=TRUE)
# display the list
my_pie
```

```
# Print the names of the list
```

```
# Retrieve the element named type
```

```
# Retrieve a truncated list
```

```
# Retrieve the element named type
```

Exploring data-sets

```
# Install packages
install.packages("openintro")
# load the package
library(openintro)
# Load package
library(tidyverse)
```

```
# Limit a glimpse of the data-set: see how the rows are stacked one below another
glimpse(loans_full_schema)
```

```
# Selecting numeric variables
loans <- loans_full_schema %>% # <- pipe operator
  select(paid_total, term, interest_rate,
         annual_income,paid_late_fees,debt_to_income)
# Use the columns stacked one below another
glimpse(loans)
```

```
# Selecting categoric variables
loans <- loans_full_schema %>%
  select() # Type the chosen columns as in the lecture slide
# Use the columns stacked one below another
glimpse(loans)
```