LBOMETR Course Book

Jem Marie M. Nario

2025-01-13

Contents

Chapter 1

Introduction

Welcome to the **LBOMETR Course Book**! This book is designed to guide students through the course by providing all necessary resources, materials, and instructions.

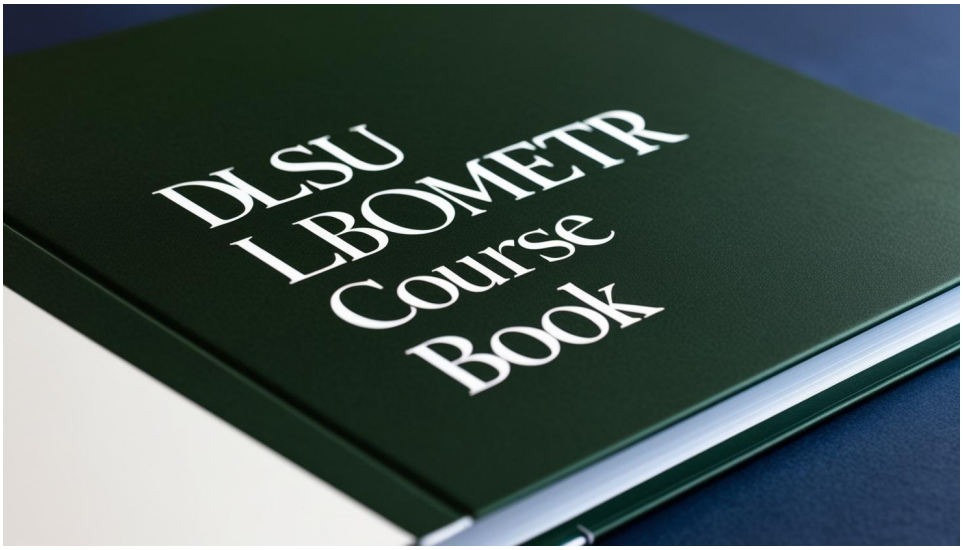

Figure 1.1: LBOMETR

This course book is intended to ensure that DLSU Carlos L. Tiu-School of Economics students will be able to learn more about Econometrics using R. You will find sections on the syllabus, course assessments, and group projects, as well as guidance for navigating the course effectively.

1.1 About Me

My name is **Jem Marie M. Nario**, and I am your lecturer for this course. I am excited to guide you through this journey of learning and discovery since I am also on a journey of learning and discovery while teaching part-time.

This book is a trial version which will be updated along the course as it also serves as a practice for me.

- **Email**: jem.nario@dlsu.edu.ph
- **LinkedIn**: linkedin.com/in/jmnario/

Feel free to reach out with any questions or concerns throughout the course.

Chapter 2

Syllabus

You can download the course syllabus using the link below:

Download Syllabus (Word Document)

2.1 Course Description

This course introduces Economics majors to more advanced commands and techniques used in the econometric software package **R**, which is commonly used in empirical research.

2.2 Learning Outcomes

2.2.1 Knowledge

- To be able to distinguish a theoretical economic model from a statistical econometric model.

- To be able to use the R software package in estimating advanced econometric models.
- To learn advanced econometric models so that students can learn new methods of research.

2.2.2 Skills

- Apply numerical and statistical techniques in economic analysis.
- Use statistical concepts as a language in economic discourse.
- Confidently write script files for economic analysis.

2.2.3 Behavior/Attitude

- To imbibe in the student the need for transparency and academic integrity when handling data analysis.
- To allow the student to learn to construct more complex programs from basic commands learned in class.

2.3 Grading

2.3.1 Grade Components

Component	Weight (%)
Attendance	5%
Group Participation	10%
Data Story Presentation	35%
Data Story Archive	50%

Component	Weight (%)
Total	100%

2.3.2 Grade Scale

Percentage Range	Grade
96 - 100	4.0
90 - 95.99	3.5
84 - 89.99	3.0
78 - 83.99	2.5
72 - 77.99	2.0
66 - 71.99	1.5
60 - 65.99	1.0

Chapter 3

Course Assessments

3.1 Data Story Archive

The **Data Story Archive** is the culmination of your group's work throughout the course. It includes your group's data story report, R script, practical assignments, and a group reflection, all compiled into a single professionally formatted PDF file.

3.1.1 Requirements

Your submission should follow this structure:

1. **Cover Page**:
    - Include the title of the Data Story, group members, and submission date.
2. **Table of Contents**:

- Provide a clear list of sections with page numbers.

3. **Data Story Report**:

   - The complete report should include:
     - **Introduction**: Problem statement and research question.
     - **Methods**: Data sources, methodology, and analysis techniques.
     - **Results**: Key findings supported by R-generated visuals.
     - **Discussion**: Implications of the findings and any limitations.
     - **Conclusion**: Summary and recommendations.
     - **Appendix**: Supporting tables, additional plots, or materials.

4. **R Script**:

   - Render your R script as an **HTML** using Quarto Markdown.
   - Ensure the script is well-structured, commented, and includes outputs like plots and tables.

5. **Computer Practicals**:

   - Include PDFs of all Quarto Markdown files from your computer practicals by printing the html as pdf.

6. **Group Reflection**:

   - Write a 1-2 page reflection on:
     - Your teamwork experience (challenges and successes).
     - What you learned from working on the data story.

- How the course contributed to your growth in data analysis and collaboration.

3.1.2 Submission

- Combine all the components into a **single PDF** file.
- Name your file as: `LBOMETR[Section_GroupNo.]_DataStoryArchive.pdf`
- **Deadline**: [11 April 2025, 21:00].
- In the event that the file is too big for Animospace, kindly submit as pdf to my email.

3.1.3 Grading Rubric for Data Story Archive

The grading rubric for the Data Story Archive is divided into three categories: **Content**, **Analysis and Technical Work**, and **Overall Presentation Quality**.

Category	Criteria	Points	Description
1. Content			

Category	Criteria	Points	Description
	Clarity of Objective	10	Clearly defined problem/question and its relevance to the course.
	Data Story Report	20	Completeness and quality of the report, including introduction, methods, results, and discussion.
	Appendix	10	Completeness of additional materials (e.g., tables, plots) in the appendix.
2. Analysis and Technical Work			

Category	Criteria	Points	Description
	R Script Quality	15	Well-structured, commented, and reproducible R script with outputs rendered as a PDF.
	Practical Assignments	15	Quality and completeness of PDFs rendered from Quarto Markdown files.
	Visualizations	15	Clear, meaningful, and well-designed plots and tables generated in R.
3. Overall Presentation Quality			

Category	Criteria	Points	Description
	Group Reflection	15	Thoughtful insights on teamwork, learning, and course experience.
	Formatting and Organization	10	Overall organization, formatting, and adherence to submission guidelines.
	Total	**100**	

3.2 Data Story Presentation

The **Data Story Presentation** is your group's opportunity to communicate your findings and insights through a live presentation. This format allows you to showcase animated visualizations and engage directly with the audience in real time. A room will be requested for you to be able to present in front of your classmates and I will be present online *hopefully this will be applicable;*

3.2.1 Requirements

1. **Objective**:
    - Your live presentation should effectively communicate your data story with clarity, engagement, and professionalism, making full use of visuals and animations to enhance understanding.
2. **Presentation Structure**: The presentation must include the following sections:
    - **Introduction**: Briefly introduce your topic, research question, and the significance of your data story (1 slide).
    - **Methods**: Provide a concise explanation of your data and analysis methodology (1-2 slides).
    - **Results**: Highlight the most important findings using R-generated visualizations, including animations if applicable (3-4 slides).
    - **Discussion and Conclusion**: Discuss the implications of your findings and conclude with actionable insights or recommendations (1 slide).
3. **Delivery**:
    - Each group member must actively participate in the presentation.
    - Presentation duration: **10 minutes**, followed by a **5-minute Q&A session**.
4. **Visualizations**:
    - Use animated or interactive visualizations (e.g., created with `gganimate` or other R packages) to effectively demonstrate key

trends and insights.

- Ensure visuals are clear, professional, and aligned with your narrative.

5. **Tools**:

   - Create your presentation using tools like Google Slides, Microsoft PowerPoint, or Canva.
   - Incorporate animated visualizations as needed.

6. **Submission**:

   - Submit your presentation slides as a **PDF file** named:

     `LBOMETR[Section_GroupNo.]_DataStoryPresentation.pdf`

   - Submit the file before your scheduled presentation time.

3.2.2 Grading Rubric

The grading rubric for the Data Story Presentation is divided into three categories: **Content**, **Visualizations**, and **Delivery and Engagement**.

Category	Criteria	Points	Description
1. Content			

Category	Criteria	Points	Description
	Introduction and Methods	10	Clear and concise introduction and explanation of methods.
	Results	20	Logical flow and depth of results, focusing on key findings.
	Discussion and Conclusion	10	Insightful discussion and actionable conclusion.
2. Visualizations			
	Quality of Visuals	20	Professional and well-designed visualizations, including appropriate use of animations.

Category	Criteria	Points	Description
	Relevance of Visuals	10	Visuals strongly support the analysis and enhance understanding.
3. Delivery and Engagement			
	Delivery	20	Confident, clear, and professional delivery by all group members.
	Audience Engagement	10	Creativity and ability to maintain audience attention.
	Q&A Session	10	Ability to effectively respond to audience questions.

Category	Criteria	Points	Description
	Time Management	10	Adherence to the 10-minute time limit and logical pacing.
	Total	**100**	

Chapter 4

Grouping Process

Students will be randomly assigned to groups of **4-5 members** based on their responses to a pre-course survey. The survey collects information that will be used to ensure fair and balanced groupings. The group assignments will be announced on the first day of the course.

4.1 Survey

Please complete the survey **before 14:30 PM on January 6, 2025** using the link:

- Google Form Survey Link

4.2 How Groups Are Formed

The groupings are created using RStudio. The coding ensures randomness while incorporating some aspects of the survey responses to balance groups.

If you wish to see the code used for grouping, you may contact me directly. However, please note: - The **CSV file with survey responses will not be shared** to protect your anonymity and privacy.

4.3 Announcement of Groups

The group assignments will be distributed on the **first day of the course**. Please check your assigned group and connect with your group members as soon as possible.

Chapter 5

Basic Introduction to R

This portion of the book offers an introduction to the basics of R. R offers a wide variety of functionality. Note that this book only offers basic Econometric analysis. It will be useful to have some basic familiarity with R and its syntax but this is not strictly necessary.

Each chapter includes both R code and results to make it easier for students to follow along, even without detailed knowledge of R.

5.1 Session Information

This version of the book was built using R version 4.4.2. See below for the session information:

```
## R version 4.4.2 (2024-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64
## Running under: Windows 11 x64 (build 22631)
```

```
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_Netherlands.utf8  LC_CTYPE=English_Netherlands.utf8
## [4] LC_NUMERIC=C                         LC_TIME=English_Netherlands.utf8
##
## time zone: Europe/Berlin
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] stringr_1.5.1 dplyr_1.1.4   bookdown_0.41
##
## loaded via a namespace (and not attached):
##  [1] vctrs_0.6.5       cli_3.6.3         knitr_1.49        rlang_1.1.4       x
##  [7] generics_0.1.3    jsonlite_1.8.9    glue_1.8.0        htmltools_0.5.8.1 t
## [13] rmarkdown_2.29    evaluate_1.0.1    jquerylib_0.1.4   tibble_3.2.1      f
## [19] lifecycle_1.0.4   compiler_4.4.2    pkgconfig_2.0.3   rstudioapi_0.17.1 c
## [25] utf8_1.2.4        tidyselect_1.2.1  pillar_1.10.0     magrittr_2.0.3    k
## [31] tools_4.4.2       cachem_1.1.0
```

5.2 Preliminaries

The first step is to gain access to R, which is free and available on the R website: http://cran.r-project.org/. Simply go to the R website, select the appropriate location and operating system, and follow the instructions to download the base distribution of R. RStudio offers a user friendly environment to run R and is recommended.

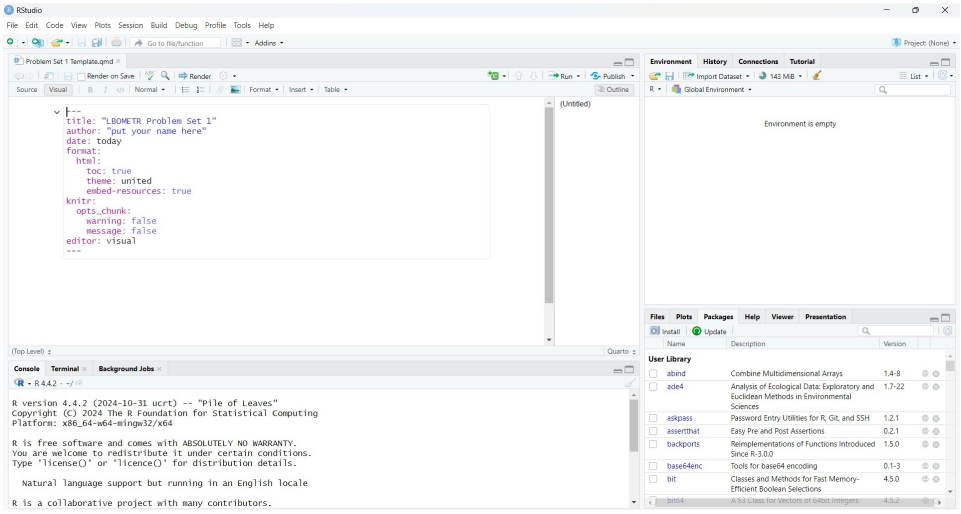

Figure 5.1: RStudio Screen

Once R is opened, we can begin to run commands. R commands can be run directly from the console, from the R script editor or from a text editor separate from R.

R offers detailed help files for each function. To access help, run:

```
?sum
```

All lines proceeded by a # are comments and will not run. For example:

```
# This is a comment. R will not recognize this as a command.
```

5.3 Quarto Markdown

In LBOMETR, Quarto Markdown will be used by the students when submitting the Scripts for the Data Story Archive. Quarto Markdown is a tool for creating documents, reports and presentations using Markdown and executable code. Below is a concise guide to help you get started, along with key shortcuts for both Mac and Windows.

5.3.1 1. Starting a Quarto File

To begin creating a Quarto document, follow these steps:

1. Open RStudio.
2. Go to **File > New File > Quarto Document.**
3. Choose the document type (e.g., HTML, PDF, Word, etc.) and specify whether the document will include code. For ease, we will use the html document type. I have also added a sample Quarto Markdown file you can copy.

   Quarto Markdown Template

5.3.2 2. Quarto Key Features

Code Chunks

Code chunks allow you to include and run code inside your document.

Inline Code

Embed R code in text using backticks and `r` .

5.3.3 Quarto Markdown Shortcuts

Action	Windows Shortcut	Mac Shortcut
Insert a new code chunk	Ctrl+Alt+I	Cmd+Alt+I
Run current code chunk	Ctrl+Shift+Enter	Cmd+Shift+Enter
Run all code chunks	Ctrl+Alt+R	Cmd+Alt+R
Run current line/selection	Ctrl+Enter	Cmd+Enter
Knit/Render document	Ctrl+Shift+K	Cmd+Shift+K
Comment/uncomment lines	Ctrl+Shift+C	Cmd+Shift+C
Insert pipe (%>%)	Ctrl+Shift+M	Cmd+Shift+M
Headings	/Number of Heading (if in Visual mode) Prefix line with #, ##, etc. manually (in Source mode)	/Number of Heading (if in Visual mode) Prefix line with #, ##, etc. manually (in Source mode)

Action	Windows Shortcut	Mac Shortcut
Bold	Ctrl+B	Cmd+B
Italic	Ctrl+I	Cmd+I
Inline code	Surround with backticks (') manually	Surround with backticks (') manually

*Note: you can choose between Source or Visual (upper left); personally, it is easier for me to use the Visual Mode compared to the Source Mode.

5.4 Packages

Each package of interest must be installed and loaded before it can be used. The packages will not be immediately available when R is opened. A package only has to be installed once on a computer, but the package will have to be loaded every time R is restarted.

We can install a package individually as we need them. For example, to install **tidyverse** and **psych**, we would do:

```
install.packages("tidyverse")
install.packages("psych")
```

In the tidyverse package, the **ggplot2** is usually included; if you do not see the package in the Packages list at the lower right, you can do this:

```
if(!("ggplot2" %in% installed.packages()[,"Package"])) install.packages("ggplot2")
```

Now that we have our packages successfully installed, we can go ahead and load them into R. Here we will load the tidyverse package as an example. We can use of all the functions available in that package once it is loaded into R. We load packages by using a `library()` function. The input is the name of the package, not in quotes.

```
library(tidyverse)
```

We can look up all of the functions within a package by using a `help()` function. For example, let's look at the functions available in the **tidyverse** package.

```
help(package = tidyverse)
```

Note that the package argument is necessary to look up all of the functions. We can also detach a package if we no longer want it loaded. This is sometimes useful if two packages do not play well together. Here we will use a `detach()` function.

```
detach(package:tidyverse)
```

For simplicity, we will assume that the reader has restarted R at the beginning of each tutorial.

5.5 Instructions for Managing Working Directories

This guide outlines how team members should set up their local working directories for collaboration, handle `.qmd` files, and organize them in a shared Google Drive.

5.5.1 1. Local Working Directory Setup

Each team member should create a local folder on their own laptops to work on `qmd` files. This folder is where you will store and edit your files before uploading them to the shared Google Drive.

5.5.1.1 Steps:

1. Create a folder on your laptop named: **DLSU_LBOMETR_Section**
2. Use this folder to save and organize your `.qmd` files while working locally.

5.5.2 2. File Naming Convention

To avoid confusion, ensure all `.qmd` files are named as follows:

- Include your name or initials and a brief description of the content
- Example:
  - `jem_nario_descriptivestatistics.qmd`
  - `jmn_piechart.qmd`

5.5.3 3. Shared Google Drive Setup

A **shared Google Drive** will serve as the central repository for all project files, including:

- `.qmd` files from all team members
- Data files
- Rendered HTML and PDF files for final submission
- Supporting documents or references.

5.5.4 4. Workflow for `.qmd` files

For each team member:

1. Work locally
    - Create your `.qmd` file in your local `DLSU_LBOMETR_Section` folder
    - Ensure it is well-documented and organized.
2. Upload to Google Drive

For the Team Leader:

1. Collect and Combine Files
    - Gather all `.qmd` files from the team folder on the shared drive.
    - Combine them
2. Render the final report

5.5.5 5. Rendering the Final Report

The final report should be rendered in HTML and printed by the team leader.

5.5.6 Summary Workflow

- **Each Team Member**:
  - Work on your `.qmd` file locally.
  - Upload your file to the shared Google Drive under `team-members-qmd`.
- **Team Leader**:
  - Collect `.qmd` files from the shared drive.
  - Combine them into a single `final_report.qmd`.
  - Render the final report into HTML and PDF.
  - Upload the rendered files to the `final-report` folder on the shared Google Drive.

This ensures an organized and efficient workflow while centralizing all files in the shared Google Drive for easy access and submission.

Chapter 6

Data Management

6.1 Where to Get Data?

Before we proceed to Data Management, let us first find where we can get data for the Data Story Archive. Note that the data you collect should still ensure that you are following the Code of Ethics and analyze Ethical Considerations.

Please view the necessary documents from the Office of the Vice Chancellor for Research and Innovation (https://www.dlsu.edu.ph/research/research-manual/)

A list of links you can search and get data from:

Note: I will not include the best links as they are pretty straightforward and these are governmental databases like the ones from World Bank, IMF, UN, Philippine Statistics Authority, and Bangko Sentral ng Pilipinas. The list here is a general list but use with proper discretion.

Name	Link	Notes
Kaggle	https://www.kaggle.com/datasets	Kaggle is where users can provide datasets; it is important to cite the sources. Mostly, datasets in Kaggle can be used for your practice.
Awesome Public Datasets	https://github.com/awesomedata/awesome-public-datasets	This repository is filled with public datasets, mostly from International contexts.
Google Dataset Search	https://datasetsearch.research.google.com/	You can download publicly available datasets from searching through Google. Though, sometimes the datasets come from 'Statista.com'. You can check the sources from the search.

6.2 Preliminaries

6.2.1 Dataset

The dataset to be used can be downloaded here: Chapter2 Practice and will be included in the Files in Animospace. The dataset was modified from the Wooldridge package in R as practice material. The particular dataset is the 'htv' dataset.

```
#install the wooldridge package. Check previous chapter on how to install packages.
load(wooldridge)
?htv #to find out about the particular dataset.
```

NOTE: The htv dataset help will tell you what the variables mean, however, for our practice, we will use the modified version of this dataset.

6.2.2 Packages

We will mostly use the `tidyverse` package, in particular, the `dplyr` package and the `tidyr` package; double-check in your Packages list whether you have these two packages; if not, you can simply install them.

6.2.3 Setting up the Directory

This is the most important step! Make sure to place the downloaded file in this folder: **DLSU_LBOMETR_Section** in your laptops. Remember, this is your local working directory. This is the working directory you choose. You can set up your working directory in the following ways:

1. Using the R Studio Menu (works for both Mac and Windows)

   a. Go to **Session > Set Working Directory > Choose Working Directory**

2. Windows:

```
# Use double backslashes `\\` or forward slashes `/`
setwd("C:\\Users\\YourUsername\\Documents")  # Example with backslashes
setwd("C:/Users/YourUsername/Documents")     # Example with forward slashes
```

3. Mac

```
# Use forward slashes `/`
setwd("/Users/YourUsername/Documents")
```

To check:

```
getwd()
```

6.2.4 Clean Everything

Do this step every time you use other data or when we do the other chapters.

```
# Remove all objects in the global environment
rm(list = ls())

# Perform garbage collection to free up memory
gc()
```

```
##           used (Mb) gc trigger  (Mb) max used  (Mb)
## Ncells 1197313 64.0    1872572 100.1  1872572 100.1
## Vcells 2800698 21.4    8388608  64.0  8388471  64.0
```

6.2.5 Importing the Dataset

We can use the `read.csv()` to load the `csv` file into R. Always call the file as something short and easily understandable. Ensure the downloaded file is in the working directory before you load the file. If the downloaded file is not located in the working directory, you will encounter issues.

I will name the file as `ch2_p1`

```
ch2_p1<-read.csv("Ch2Practice.csv")
```

We can use the `head()` function to inspect the first six rows of the dataset:

```
head(ch2_p1)
```

```
##        WAGE  ABILITY EDUCATION NORTHEAST NORTHCENTRAL WEST SOUTH EXPERIENCE MOTHEREDUC
## 1 12.019231 5.027738        15        no           no  yes    no          9         12
## 2  8.912656 2.037170        13       yes           no   no    no          8         12
## 3 15.514334 2.475895        15       yes           no   no    no         11         12
## 4 13.333333 3.609240        15       yes           no   no    no          6         12
## 5 11.070110 2.636546        13       yes           no   no    no         15         12
## 6 17.482517 3.474334        18       yes           no   no    no          8         12
##   SIBLINGS URBAN X18INNORTHEAST X18INNORTHCENTRAL X18INSOUTH X18INWEST X18INURBAN X17TU
## 1        1   yes              1                 0          0         0          1   7.5
## 2        4   yes              1                 0          0         0          1   8.5
## 3        2   yes              1                 0          0         0          1   7.3
## 4        1   yes              1                 0          0         0          1   9.4
## 5        2   yes              1                 0          0         0          1   7.3
## 6        2   yes              1                 0          0         0          1   7.3
```

```
##   EXPER.2
## 1      81
## 2      64
## 3     121
## 4      36
## 5     225
## 6      64
```

6.3 Data Cleaning

As you can see, there are 22 columns. Let's simplify by only choosing the following: WAGE, URBAN, X17TUITION, X18TUITION and EXPER.2. We can do this using the `select()` function in `dplyr`. We will save them into a new data frame, `ch2_p1.1.`

```
library(dplyr)
```

```
ch2_p1.1<-select(ch2_p1, #the original dataset
       WAGE, URBAN, X17TUITION, X18TUITION, EXPER.2)
```

6.3.1 Renaming the Variables

We will edit the names to much easier conventions. First, let us say that we just want to change them to lowercase names.

```
names(ch2_p1.1)<-tolower(names(ch2_p1.1))
```

Inspect:

```
head(ch2_p1.1)
```

```
##        wage urban x17tuition x18tuition exper.2
## 1 12.019231   yes   7.582914   7.260242      81
## 2  8.912656   yes   8.595144   9.499537      64
## 3 15.514334   yes   7.311346   7.311346     121
## 4 13.333333   yes   9.499537  10.162070      36
## 5 11.070110   yes   7.311346   7.311346     225
## 6 17.482517   yes   7.311346   7.311346      64
```

Let us change the names of `x17tuition, x18tuition,`and`exper.2` to the names similar to what is found in the 'htv' dataset: `x17tuition` to `tuit17`, `x18tuition` to `tuit18` and `exper.2` to `expersq` . To do this, we will use the `rename()` in `dplyr`. We will also use the (`%>%`) for this.

```
ch2_p1.1<-ch2_p1.1 %>%
  rename(
    tuit17 = x17tuition,
    tuit18 = x18tuition,
    expersq = exper.2
  )
```

Inspect again:

```
head(ch2_p1.1)
```

```
##        wage urban   tuit17    tuit18 expersq
```

```
## 1 12.019231   yes 7.582914  7.260242      81
## 2  8.912656   yes 8.595144  9.499537      64
## 3 15.514334   yes 7.311346  7.311346     121
## 4 13.333333   yes 9.499537 10.162070      36
## 5 11.070110   yes 7.311346  7.311346     225
## 6 17.482517   yes 7.311346  7.311346      64
```

6.3.2 Sorting certain values

Let's say, we want to arrange `wage`. We will create a different data for this. We use `arrange` in `dplyr` package.

```
ch2_p1sort<-arrange(ch2_p1.1,
                    wage)
```

Inspect:

```
head(ch2_p1sort)
```

```
##       wage urban    tuit17    tuit18 expersq
## 1 1.023529   yes  2.088957  2.251239     169
## 2 1.073345    no  7.520245  7.520245     196
## 3 1.102362   yes  7.355460  6.922371     196
## 4 1.250000   yes  9.417682  8.826549     100
## 5 1.373626   yes  9.692757  9.692757     225
## 6 1.442308    no 11.280367 11.280367      NA
```

What if you have this kind of data?

```
head(ch2_p2)
```

```
##    ch2_p2
## 1 $10,000
## 2 $20,500
## 3 $15,250
## 4 $30,000
## 5 $50,750
```

Let's sort this:

```
ch2_p2sort<-arrange(ch2_p2)
head(ch2_p2)
```

```
##    ch2_p2
## 1 $10,000
## 2 $20,500
## 3 $15,250
## 4 $30,000
## 5 $50,750
```

It did not work. The problem is, `ch2_p2` is not numeric. We can check:

```
class(ch2_p2$ch2_p2)
```

```
## [1] "factor"
```

We need to make it into a numeric value but we have a , and $. We need to remove them. We use the `str_replace` function in the `stringr` package.

```
library(stringr)
```

```
ch2_p2$ch2_p2<-str_replace(
  ch2_p2$ch2_p2, #column we want to edit
  pattern = ',', #what to find
  replacement = '' #what to replace it with
)
```

```
head(ch2_p2)
```

```
##   ch2_p2
## 1 $10000
## 2 $20500
## 3 $15250
## 4 $30000
## 5 $50750
```

Now, let us remove the dollar sign; usually, simply doing the same thing we did with the comma works, but, there are some symbols that are used as "special character". To "force" R to replace the presence of '$', we add two backslashes before the dollar sign.

```
ch2_p2$ch2_p2<-str_replace(
  ch2_p2$ch2_p2,
  pattern = '\\$',
```

```
  replacement = ''
)
```

Can you inspect it on your own?

Simply type the code in the empty code chunk then run it by pressing **Ctrl+Enter** or **Cmd+Enter**

Now, sort `ch2_p2`

```
ch2_p2sort<-arrange(
  ch2_p2,
  ch2_p2
)
```

```
head(ch2_p2sort)
```

```
##   ch2_p2
## 1  10000
## 2  15250
## 3  20500
## 4  30000
## 5  50750
```

We can see that it was arranged, however, take a look at the way ch2_p2 was encoded; it is not numeric. So, we need to change this.

```
class(ch2_p2$ch2_p2)
```

```
## [1] "character"
```

Change to numeric through `as.numeric()`

```
ch2_p2$ch2_p2<-as.numeric(ch2_p2$ch2_p2)
```

Inspect on your own:

6.3.3 Pipe Operator

`%>%` allows functions to be chained; it can be read as "then" - it tells R to do whatever comes after it to the stuff that comes before it.

6.3.4 Adding columns

We will be using the pipe operator and the `mutate` to add a new column to `ch2_p1.1` based from details found in `ch2_p1`, particularly, NORTHEAST, NORTHCENTRAL, WEST, and SOUTH. We will call this new column as `location`

```
ch2_p1.1<-ch2_p1.1 %>%
  mutate(
    location = case_when( #creates conditional statements
      ch2_p1$NORTHEAST == "yes" ~ "northeast", #If NE is "yes", location is "nor
    ch2_p1$WEST == "yes"~"west", #If WEST is "yes", location is "west"
    ch2_p1$NORTHCENTRAL == "yes"~ "northcen",
```

```
      ch2_p1$SOUTH == "yes"~ "south",
    TRUE~"other")
  )
```

Inspect the data:

Now I want you to create a new column called, `tuit_diff` wherein it is the difference between `tuit18` and `tuit17`. In this case, there is no need to use `case_when` since there is no conditional statements to be used. It is straightforward that you simply need to subtract `tuit17` from `tuit18`. You will need to use `mutate` still. How will you create that?

6.3.5 Transforming values

Now, you can see that urban is a `character` that is "yes/no". We need to change that to numeric value. This is particularly useful when we use dummy variables later on. We will not use `case_when` as it is not necessary; rather, we will use `ifelse`:

```
ch2_p1.1<-ch2_p1.1 %>%
  mutate(
    urban = ifelse(urban=="yes", 1,0) #replace "yes" with 1 and "no" with 0
  )
head(ch2_p1.1) #default is first 6 rows
```

```
##        wage urban   tuit17    tuit18 expersq  location
## 1 12.019231     1 7.582914  7.260242      81      west
## 2  8.912656     1 8.595144  9.499537      64 northeast
```

```
## 3 15.514334     1 7.311346  7.311346     121 northeast
## 4 13.333333     1 9.499537 10.162070      36 northeast
## 5 11.070110     1 7.311346  7.311346     225 northeast
## 6 17.482517     1 7.311346  7.311346      64 northeast
```

Now, I want you to create groups for `expersq`. `NA` should now be 0, assign 1 if less than 50, assign 2 if between 50 and 100, assign 3 if between 100 and 200, and for the rest, assign 4.

Clue: conditional statements like between 50 and 100 should be like this:

```
values>=50 & values < 100
```

Your answer should look like this:

```
##        wage urban   tuit17    tuit18 expersq  location
## 1 12.019231     1 7.582914  7.260242       2      west
## 2  8.912656     1 8.595144  9.499537       2 northeast
## 3 15.514334     1 7.311346  7.311346       3 northeast
## 4 13.333333     1 9.499537 10.162070       1 northeast
## 5 11.070110     1 7.311346  7.311346       4 northeast
## 6 17.482517     1 7.311346  7.311346       2 northeast
```

6.3.6 Summarizing

Let us get the average of wages by location, which we'll call `ave.wage`, by using the `group_by()` and `summarise()` functions in dplyr

```
ch2_p1.1ave<-ch2_p1.1 %>%
  group_by(location) %>% #group by location, THEN
  summarise(ave.wage=mean(wage)) #calculate the mean of wages for each location
head(ch2_p1.1ave)
```

```
## # A tibble: 4 x 2
##   location  ave.wage
##   <chr>        <dbl>
## 1 northcen      12.5
## 2 northeast     15.0
## 3 south         12.4
## 4 west          14.1
```

Say that you want to see the average wage in the south area. We can do this by using `filter()`

```
ch2_p1.1ave %>% filter(location=="south")
```

```
## # A tibble: 1 x 2
##   location ave.wage
##   <chr>       <dbl>
## 1 south        12.4
```

How would you sort the dataset by average wage, from highest to lowest? Now you see that it is arranged alphabetically, so how will you arrange it?

6.3.7 Merging datasets

We have two main datasets, `ch2_p1.1` and `ch2_p1.1ave`. By doing this, we could compare side-by-side each observation compared to the average per location.

We will join the datasets by `location` variable, since that is consistent across both datasets. We name the new file as `ch2_p1merged`:

```
ch2_p1merged<-merge(x=ch2_p1.1, y=ch2_p1.1ave, by="location")
head(ch2_p1merged)
```

```
##   location      wage urban   tuit17   tuit18 expersq ave.wage
## 1 northcen 15.294118     1 8.334936 8.334936       3 12.54078
## 2 northcen 17.006804     1 8.334936 8.334936       0 12.54078
## 3 northcen  3.755868     1 8.334936 8.334936       3 12.54078
## 4 northcen  5.288462     1 6.742574 7.198132       2 12.54078
## 5 northcen  9.072165     1 7.305873 7.356897       0 12.54078
## 6 northcen  9.384164     1 8.334936 8.334936       3 12.54078
```

6.3.8 Splitting datasets

Say I want to save different datasets based on the `location` column.

```
northcen_data<-ch2_p1.1 %>% filter(location=="northcen")
```

You can save it as a `.csv` file:

```
write.csv(northcen_data, "northcentral.csv", row.names = FALSE)
```

Can you do the others?

6.3.9 Dates

We are going to work on dates when we move to the next chapter but, here is something initial and necessary.

```
##   ID date_of_birth
## 1  1    15-05-1990
## 2  2    20-08-1985
## 3  3    01-12-2000
## 4  4    10-03-1995
## 5  5    25-07-2010
```

```
## 'data.frame':    5 obs. of  2 variables:
##  $ ID           : int  1 2 3 4 5
##  $ date_of_birth: chr  "15-05-1990" "20-08-1985" "01-12-2000" "10-03-1995" ...
```

To convert the `character` format to date format, we do this:

```
date_data$date_of_birth <- as.Date(date_data$date_of_birth, format = "%d-%m-%Y")


head(date_data)
```

```
##   ID date_of_birth
## 1  1    1990-05-15
```

```
## 2  2    1985-08-20
## 3  3    2000-12-01
## 4  4    1995-03-10
## 5  5    2010-07-25
```

Say you want to calculate the age:

```
date_data$age <-as.numeric(floor((Sys.Date()-date_data$date_of_birth)/365.25))
head(date_data)
```

```
##   ID date_of_birth age
## 1  1    1990-05-15  34
## 2  2    1985-08-20  39
## 3  3    2000-12-01  24
## 4  4    1995-03-10  29
## 5  5    2010-07-25  14
```

6.3.9.1 Custom Reference Date:

```
ref_date<-as.Date("2020-01-01")
date_data$age2<-as.numeric(floor((ref_date-date_data$date_of_birth)/365.25))
head(date_data)
```

```
##   ID date_of_birth age age2
## 1  1    1990-05-15  34   29
## 2  2    1985-08-20  39   34
## 3  3    2000-12-01  24   19
```

```
## 4  4    1995-03-10  29   24
## 5  5    2010-07-25  14    9
```

6.4 Next Meeting

6.4.1 Topics:

1. Feedback on the Practical
2. Time-Series Data and Panel Data - Data Management
   a. Including transforming dataset from long to wide and v.v.
3. Missing values

Chapter 7

Data Management Practical

In your Quarto Markdown files, you need to answer the following questions. Answers will be given the week after the Practical as a form of Feedback. You can answer by group. It would be great to include which group member did what.

In the first part of the practical, answer the following reflection:

1. Do you think you were able to input correctly all the codes in the empty code chunks? Why do you think so? What did you find difficult?

Now comes the practical proper:

Using the `kpop idols` dataset which you download: Practical 1 - also made available in Animospace, answer the questions.

Ensure that you can render individually before uploading in your shared Google Drive. Failure to render the file means you were unable to do the

Practical. The leader must take note of this since accomplishing the practicals are part of your grade in Group Participation and the Data Story Archive.

1. Separate the dataset into two datasets: one for `males` and one for females. Save these datasets as `males.csv` and `females.csv`
2. Edit the column names for both `males` and `females` datasets to make them shorter, easier to understand, and consistent
3. Remove the following columns from both datasets: Instagram, Korean Name, K.Stage Name, Stage Name
4. How many males and females are in the dataset? *Hint:* `nrow()`
5. Create a binary variable `not_seoul` for both datasets.
    1. Assign `1` if `birthplace` is **not Seoul,** and `0` otherwise
    2. Count how many individuals are from Seoul and how many are not for both datasets.
6. How many males are eligible for military service (ages 18-28 as of February 27, 2024)?
    1. Filter the `males` dataset for this age range, how many from the Country of South Korea (only filter according to Country for straightforwardness) and count how many qualify.
7. Assign generations based on age (as of June 29, 2023 when the Korean Age system was abolished) for both datasets.
    1. Generation criteria:
        1. 1st Gen: Age>=40
        2. 2nd Gen: 31<=Age<=39

3. 3rd Gen: 25<=Age<=30
4. 4th Gen: Age<=24

Count the number of individuals in each generation for males and females

8. Create a new column `income` for both datasets using hypothetical values based from your hypothesis on age influencing income levels of idols. Explain first what your hypothesis is - are idols who are older earning more or less? Why or why not? Also think if you believe females earn more than males or vice versa?

    1. Use `set.seed()` for reproducibility and generate random income values.

        1. Please have different income values depending on their age. You can do similar groupings as Step 7 for this.

    2. Compare the mean income

9. Combine the `males` and `females` datasets
10. Calculate the income difference between males and females

    1. Create a column `income_diff` to calculate how much more or less each individual's income is compared to the average income of the other gender.

11. Save the final dataset named **Ch2_Practical_Section_GrpNo.csv**

7.1 End!