In-Class Lab 11

ECON 4223 (Prof. Tyler Ransom, U of Oklahoma)

October 5, 2021

The purpose of this in-class lab is to use R to import data. You don't need to turn this in on Canvas, but I am hopeful that this will be a useful reference for you.

For starters

First, install the `ipumsr` package. This package is useful for understanding how to import data into R, and also how to use data products from the University of Minnesota Population Center (a.k.a. IPUMS).

Open up a new R script and add the usual "preamble" to the top:

```
# Add names of group members HERE
library(tidyverse)
library(magrittr)
library(modelsummary)
library(ipumsr)
```

Types of data

Data can come in a variety of forms. The most common are:

- Comma Separated Values (CSV)
- Fixed-width files (DAT)
- Microsoft Excel spreadsheets (XLSX)
- Tab Separated Values (TSV)
- Other delimited values (e.g. semicolon-separated values, slash-separated values, etc.)
- Other formats (JSON, etc.)

How to import these types of data

In RStudio, just click "Import Dataset" on the "Environment" pane (usually in the top right hand corner of RStudio). This will open a window where you can select the file location, how the dataset is delimited, etc. In the bottom-right of this new window, it will generate code for you that will reproduce this action. **I strongly recommend you include that code in your R script moving forward.**

Working directory

RStudio can be tricky if you don't have your working directory appropriately set. You need to put your data in a place where RStudio can find it.

You can set your working directory by typing `setwd("path/to/my/directory")` or by clicking "Session" -> "Set Working Directory" and then following the prompt.

Importing data from a URL You can also directly import data from a URL, if your data exists on the web somewhere. This can be convenient if you don't want to store the raw data on your personal computer.

```
df.auto <- read_csv('https://tyleransom.github.io/teaching/MetricsLabs/auto.csv')
```

Example (from lab 9):

Using IPUMS data

To use data from any of the IPUMS sources (Current Population Survey, American Community Survey, etc.), you need the following four things:

- Install and load the `ipumsr` package
- Download the .DAT raw data file form the IPUMS data extracts page
  - Note: it will download as a `.dat.gz` file
- Download the "DDI" codebook file from the IPUMS data extracts page
  - Note: you may need to right-click and say "save link as"
- Copy and past the "R" command file from the IPUMS data extracts page

The R command file is reproduced below:

```
# NOTE: To load data, you must download both the extract's data and the DDI
# and also set the working directory to the folder with these files (or change the path below).

if (!require("ipumsr")) stop("Reading IPUMS data into R requires the ipumsr package. It can be installe

ddi <- read_ipums_ddi("cps_00010.xml") # filename may be different for you!
cps_data <- read_ipums_micro(ddi)
```

Need to work with value labels before you explore the data

```
cps_data %<>% mutate(educ = as_factor(lbl_clean(EDUC)))
cps_data %<>% mutate(sex  = as_factor(lbl_clean(SEX)))
cps_data %<>% mutate(age  = as.numeric(AGE))
```

Now Explore the Data (e.g. Lab 1, Lab 2)

```
cps_data %>% select(age,educ,sex) %>%
             datasummary_skim(histogram=F)
cps_data %>% select(age,educ,sex) %>%
             datasummary_skim(type="categorical",
                              histogram=F)
```

Using value labels

IPUMS data in the `ipumsr` package comes pre-labeled, which is a really nice feature. For example, we can do something like

```
ipums_val_labels(cps_data$SEX)
```

which tells us that 1 corresponds to "Male" and 2 to "Female" with 9 being "NIU" (or "Not In Universe," another way of saying "NA").

For more examples of what the `ipumsr` package can do, check out https://cloud.r-project.org/web/packages/ipumsr/vignettes/ipums.html.

Using a Codebook

When using survey data not from IPUMS, you will typically have to figure out the labels yourself:

- Typically, data from surveys will have codes for categorical variables (e.g. sex, race/ethnicity, education level, etc.)
- You will need to find the codebook on your own and figure out which codes correspond to which values
- Often, there are "NA" codes (usually weird values like 99, 999, -99, etc.). Be on the lookout for these!

Loading data from other packages

In Lab 10, I introduced you to the `pdfetch` package. I will now show you how to import stock price data and FRED data using this package. Note that there are other packages out there that will also do this, so feel free to look around.

```
library(pdfetch)
```

```
## Warning: package 'pdfetch' was built under R version 4.1.1
```

```
library(tsibble)
```

```
## Warning: package 'tsibble' was built under R version 4.1.1
```

```
##
## Attaching package: 'tsibble'
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, union
```

```
# get data on GDP and consumption expenditures from FRED
df <- pdfetch_FRED(c("GDPC1", "PCECC96")) # N.B. this is an object of type XTS, not tsibble

# get Apple & Amazon stock price data from Yahoo Finance
df <- pdfetch_YAHOO(c("aapl","amzn"), fields=c("adjclose","volume"))
```

Exporting summary stats and regression tables from `modelsummary` to Word

You can also use `modelsummary()` and `datasummary_skim()` to easily export directly to Microsoft Word.

Now Explore the Data (e.g. Lab 1, Lab 2)

```
cps_data %>% select(age,educ,sex) %>%
             datasummary_skim(type="categorical",
                              histogram=F,
                              output = "my_word_doc.docx")
cps_data %>% select(age,educ,sex) %>%
             datasummary_skim(histogram=F,
                              output = "my_2nd_word_doc.docx")
```