IHDS Project Workbook

Zizun Zhou

2022-08-31

Coding Project 1: Exploring IHDS Data

Introduction

For this coding project, we will be using household-level data from the India Human Development Survey, 2005 (IHDS-I) and the India Human Development Survey, 2011-12 (IHDS-II). Both surveys involve face-to-face interviews with representative rural and urban households in India. The IHDS is a collaborative project between four institutions: the University of Maryland, College Park; the National Council of Applied Economic Research (NCAER) in Delhi; Indiana University; the University of Michigan.

This project will introduce you to the basics of using R without going into the details of R programming. Here are some things that will be covered:

1. How to set up your R working environment
2. How to read in data
3. How to plot data
4. How to select and filter data

If you are already familiar with certain tasks above, feel free to skip them or do something more challenging in lieu of the tasks.

Task 1: Setting up your R environment

Congratulations, we have done most of the setup work for you already!

Create a new R script, name it "firstname_lastname_ihds.R", and save it under "dev_econ_cohort_2022\code\workbook".

Copy the following code and place it at the start of every R script and R markdown document that you create for the cohort:

```r
################################################################################
################################################################################

# COPY AND PASTE EVERYTHING HERE TO EACH CODE FOR CONSISTENCY
# DO NOT EXCEED WIDTH OF HASHTAGS (80 CHARACTERS)

# clear global environment
rm(list = ls(), envir = globalenv())

# clear memory
gc()
```

```r
# load required libraries
require(tidyverse)
require(data.table)
require(ggplot2)
require(testthat)
require(janitor)
require(rprojroot)

# load directories
homedir <-
  find_root(criterion = has_file("dev_econ_cohort_2022.Rproj"))
codedir <- file.path(homedir, "code")
setwd(codedir)
source("paths.R")

############################################################################
############################################################################
```

There are three parts to the above code. First, it removes all unwanted objects from your working environment and cleans the memory, which helps if you want to switch between large projects without restarting R. Secondly, we load several packages that are useful for a variety of R projects. Thirdly, we specify the path to your GitHub repository, which is where you will be making edits to your code.

Task 2: Read in data

Next, we will read in the raw data, which is conveniently stored in R Data File (RDA) format.

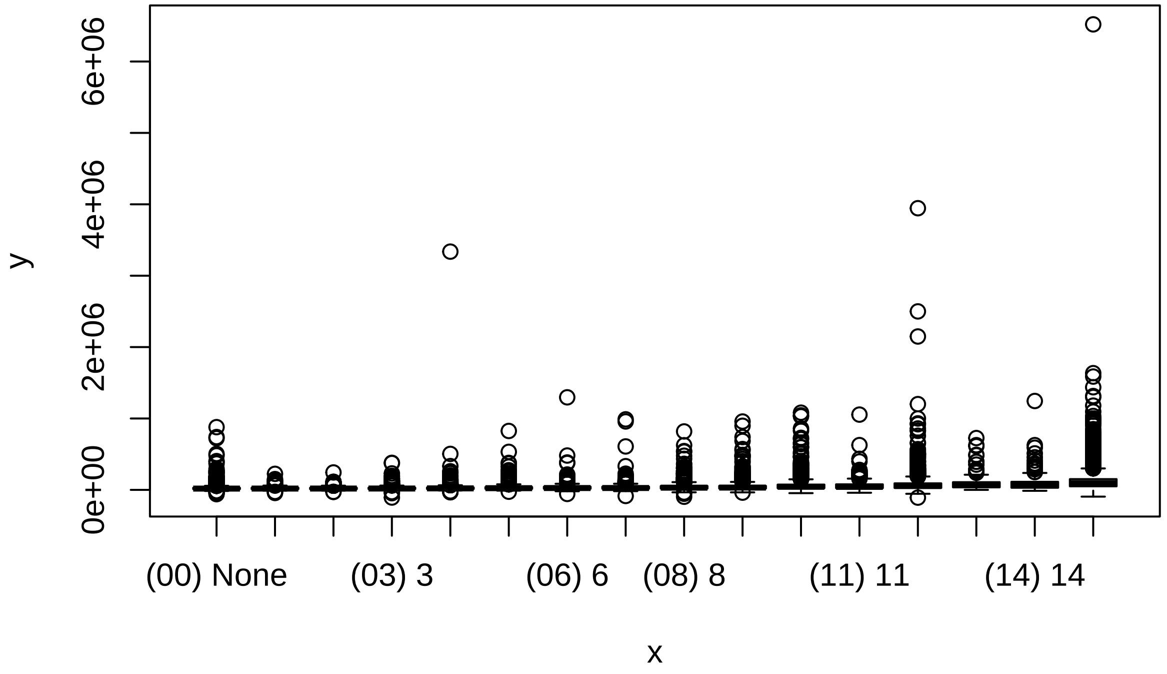
```r
# import data
file.path(prelimdir, "ihds1", "DS0002") %>% setwd()
ihds1_filename <- dir() %>% str_subset(".rda")
file.path(getwd(), ihds1_filename) %>% load(envir = globalenv())
tmpname <- ls() %>% str_subset("da2")
ihds1 <- tmpname %>% get()
rm(list = c(tmpname, "tmpname", ihds1_filename))
```

The above snippet of code shows how you can load the IHDS-I data. Try to figure out what each part of the code does, and then replicate the code to read in IHDS-II. The code above shows a slightly elaborate way of reading in data because it reduces the amount of hard-coding that is done, which is a good habit to get into. By hard-coding, I mean values (numbers or characters) that you manually specify in your. In general, you want to minimize the number of these values, since they can make it much more difficult to maintain your code. Also, remove redundant objects in your global environment as you code. This is also good practice.

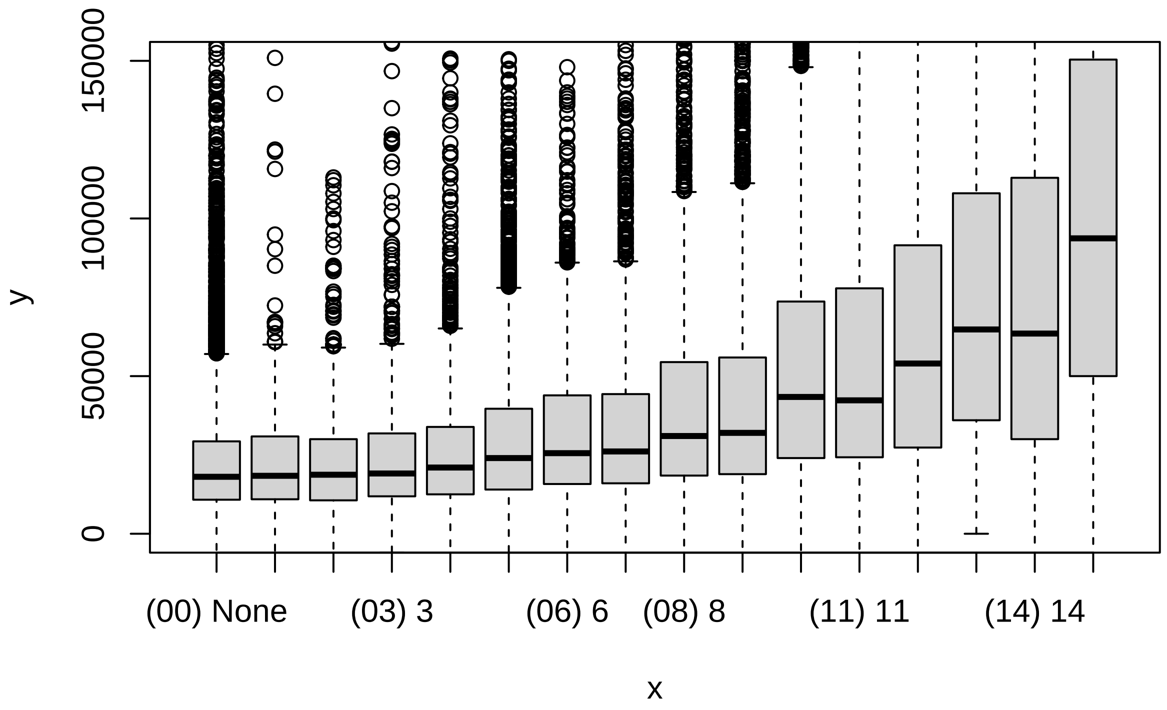
From this point onward, I will demonstrate coding using IHDS-I data, while your task will be performed on IHDS-II data.

Task 3: Plot data

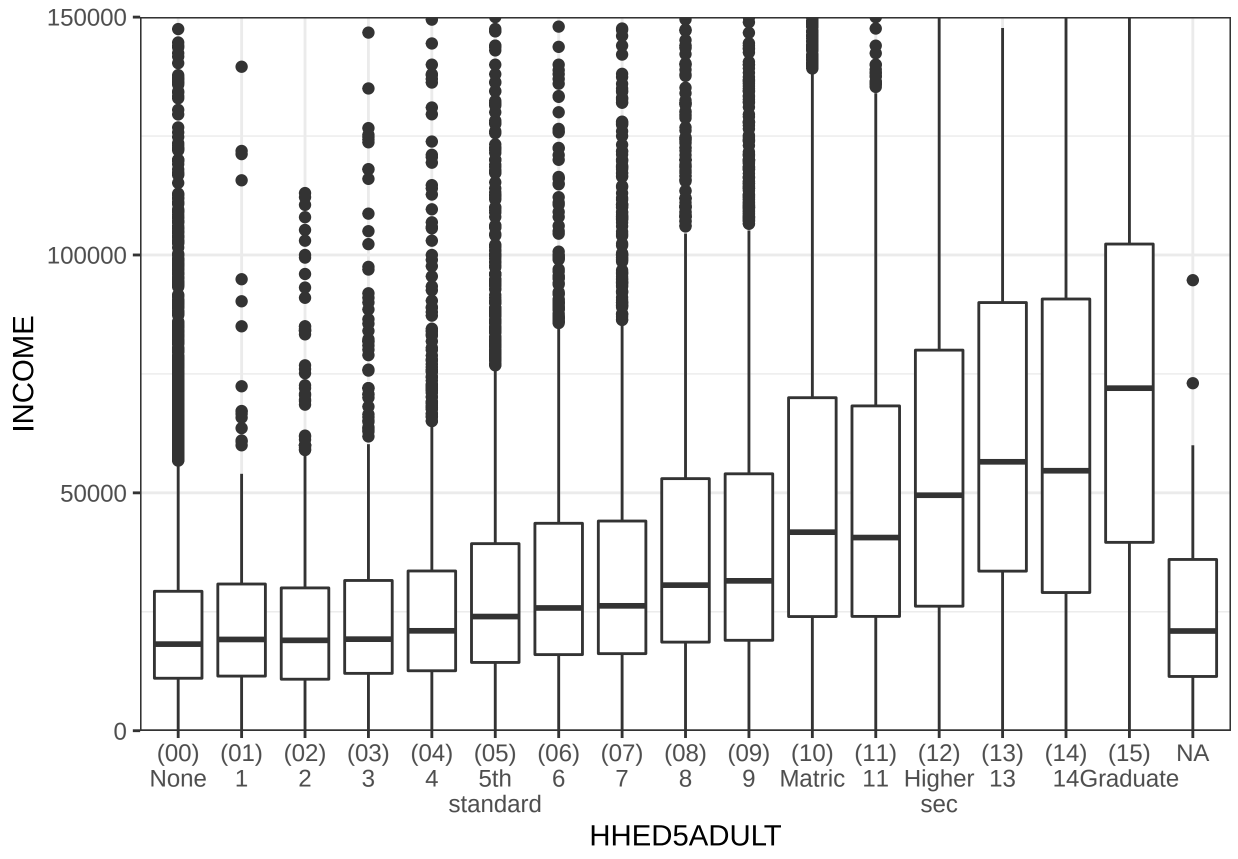
Now let us do some plotting, which R is very useful for. Suppose we wanted to plot household income against the highest education received by an adult in the household. Try the following code:



As you can see, when you do a simple plot(x-varibale, y-variable), R will try to fit all data points onto your graph. This can be problematic for data that has extreme values, which is what we see on our graph. Normally, the way to resolve this problem is to transform the data and exclude outliers. Since you will not learn how to do this until the next section, here is a simple fix by tweaking the display range of your axes:



Specifying the axis limits manually has made our data more visible. The plot function is convenient, since it is often able to select the most appropriate type of graph for your data. However, if you want more customization for your graph, you should consider using the *ggplot2* package. This powerful package requires a separate tutorial, so I will simply show you an example of what it looks like:



Now it is your turn. Use the plot function to plot the household's school/college fees (CO35) against the household's total consumption (COTOTAL). For those of you who find this too easy, try to use ggplot2 to create a scatterplot where regions with greater concentrations of data look darker.

Task 4: Select and filter data

Your full data table could turn out to be very big, in which case you want to avoid repeatedly referencing the entire table. Selecting particular variables and subsetting rows based on conditions is the easiest way to do this.

You can select data using just base R. However, it is much more common to use the *dyplr* package that comes with *tidyverse* or the *data.table* package. Here, I will show you how to select variables and subset rows using dyplr.

For any subset of data, we usually want to preserve columns with identification information. Doing so will allow us to easily retrieve other columns for our subset should we want to do so later on. For IHDS1, we want the *IDHH* column and possibly a few other ID columns for geographical information. I also want to look at one particular variable, namely the dummy variable *HHED2* that indicates whether or not any adult in the household is literate. Here's a demonstration of how I would retrieve the household literacy data for the state of Jammu and Kashmir:

```
# select variables and subset rows
littable <- ihds1 %>%
  select(IDHH, STATEID, DISTID, HHED2) %>%
  filter(STATEID == "(01) Jammu & Kashmir")
```

You should save your subset as a new object using the assign arrow in order to access it later. Following the above example, try to subset total consumption expenditure in IHDS-II by caste (*ID13*).