SR275 Cleaning (Palmy)

2023-01-14

Summary

The goal of this data cleaning process for SR275 dataset is to have one row PER teacher PER school year. There are multiple duplicates that we need to take care of. Here are the steps that I follow in order to clean the data:

1. Check for duplicates (I write several R functions for this)
2. Narrow the data to only the years with duplicate rows
3. Come up with an idea of how to remove duplicates
4. Repeat

The original dataset has 11 million rows.

Codebook: https://docs.google.com/spreadsheets/d/12980A-8Q8JW_sZlAnGsHNtWILRCS3Zcv/edit#gid=1315477372 (https://docs.google.com/spreadsheets/d/12980A-8Q8JW_sZlAnGsHNtWILRCS3Zcv/edit#gid=1315477372)

Preparation

This is a medium size dataset (~5 G) so I am going to load the libraries that help me import the data and maximize my computer's working power.

```
library(readstata13)
library(tidyverse)

#maximize the cores for faster processing since this is a large dataset
#install.packages("doMC")
library(doMC)
registerDoMC(cores = 4)

#load the data - use the stata13 command because the file is stata version - 11 milli
on rows
dat <- read.dta13("~/Downloads/INDEP/teacher_attrition_covid/data_S275_aesy_01_1995-9
6to2020-21prelim.dta")
```

Functions

There are three functions that I wrote:

1. dupecol: this function creates (or update) a variable called “dupyear” and attach it to the dataset in order to track whether there are duplicates within certification number within school year.
2. anydupe: this function checks if the data contains school years with duplicates, does NOT contain school years with duplicates, or contains both years with and without duplicates. I use this to check the data
3. narrow: this function narrow down the dataset (and make it smaller) by only selecting the years with duplicates. This helps me pinpoint potential decision for removing duplicates. I keep repeating this process until there are no more duplicates within a certification number within a school year.

The codes for these functions are:

```
dupecol <- function(data) {
  data <- data %>% group_by(cert) %>% mutate(dupyear = case_when(
    length(schyear) != length(unique(schyear)) ~ "Contains dupes",
    length(schyear) == length(unique(schyear)) ~ "Do not contains dupes"
  ))
  return(data)
}

anydupe <- function(data) {
  data <- data %>% group_by(cert) %>% mutate(dupyear = case_when(
    length(schyear) != length(unique(schyear)) ~ "Contains dupes",
    length(schyear) == length(unique(schyear)) ~ "Do not contains dupes"
  ))
  return(unique(data$dupyear))
  #return(data)

}

narrow <- function(data) {
  data <- data %>% filter(dupyear == "Contains dupes")
  return(data)
}
```

Data Cleaning

Keep in mind that the starting data has 11 million rows. The following decisions can be done right of the batch because they are pretty obvious.

1. Remove all rows in the original data where major is NOT equal to 1.
2. Select appropriate department root (droot).
3. Drop rows that are missing certification number (cert)

```
#load the data - use the stata13 command because the file is stata version - 11 milli
on rows
dat <- read.dta13("~/Downloads/INDEP/teacher_attrition_covid/data_S275_aesy_01_1995-9
6to2020-21prelim.dta")


#create a new data file and start working on it
data <- dat %>% filter(major == 1) #select those rows that have major = 1 (which we a
ssumes mean that it is the main observation of a person). Now we are down to 3.3 mill
ion rows

#create teacher variable, 1 if teacher, 0 if not
data <- data %>% mutate(teacher = case_when(
  droot == 31 ~ 1,
  droot == 32 ~ 1,
  droot == 33 ~ 1,
  droot == 34 ~ 1,
  TRUE ~ 0
))

#drop if missing certification
sum(!complete.cases(data$cert)) #no NA - so there might be some other type of missing
data
data <- data %>% filter(cert != "") #drop empty string - we are now down to 3.2 milli
on rows

#arrange data by cert so that we can make decisions to remove duplicates
data <- arrange(data, cert)
head(data$cert, 1e4)

#perform our duplicate checks
data <- dupecol(data)
anydupe(data)
table(data$dupyear) #this line of code is like tab in Stata. It gives out how many ro
ws are NOT duplicates within cert and within year; as well as how many rows have dupl
icates within cert and within year.
```

After all the steps above, the data now has 3.2 million rows. Using table() function above, the data has only 81020 rows that contain duplicates. So I am going to narrow down by create a new data set called “dsub” which is a subset of data that only contains the rows with duplicates. I use the function narrow() that I created above. **This is just the way for me to dive deeper into the data and test my hypothesis, once I figure out what exactly I have to do to remove duplicates, I will perform it on the original data set.**

```
dsub <- narrow(data)

anydupe(dsub) #this obviously results in ALL rows having duplicates

#we want to check if the anydupe() function is performing well.
#we will also subset the data with NO dupes to double check (3 million something row)
dsub_nodupe <- data %>% filter(dupyear == "Do not contains dupes")

#check with the same test
anydupe(dsub_nodupe) #YAY no duplicates; so my manual function works
```

Working with the subset data “dsub” which has 81000 something rows, I am going to add these steps in an effort to remove duplicates. **In between these steps, I constantly use the function anydupe() and narrow() to dive deeper into the section of the data that contains duplicates:**

4. Check if there are more than 1 droot number per one certification (NOT per school year). If so
5. If there are multiple droot number, that means I will need to look closely to those who might have switched between teaching jobs, or to and from teaching to non-teaching jobs. (we want to keep these). Check if there are anybody who has NEVER been a teacher and remove them.
6. I am going to remove dupliocated rows within a certification number within school year that lower certificate FTE than the maximum FTE that a person has. Maximum FTE signifies that this particular row is corresponded to this person’s job within this school year.

```
#check if there is only 1 droot number PER certificate number within this data.
dsub <- dsub %>% group_by(cert) %>% mutate(checkdroot = max(droot)) #created a variab
le to check droot
identical(dsub$droot, dsub$checkdroot) #this line of code checks whether the droot va
riable and the newly created "checkdroot" variable is identical
#the results in FALSE - meaning that there are some certification with multiple droot
s

#remember that we have the "teacher" variable which is derived from certain droots nu
mber that are assumed to be associated with being a teacher.
#So we perform the same test with the variable teacher
dsub <- dsub %>% group_by(cert) %>% mutate(everteach = max(teacher)) #create a variab
le to compare with teacher variable
identical(dsub$teacher, dsub$everteach) #this results in FALSE; meaning that there ar
e people who are never a teacher so we need to filter that out

dsub <- dsub %>% filter(everteach != 0) #drop those that NEVER were teachers; we are
down to 53000 something rows

#check for dupes
anydupe(dsub) #still contains dupes, so I am going to move on to FTE

dsub <- dsub %>% group_by(cert, schyear) %>% mutate(maxfte = max(certfte))
identical(dsub$certfte, dsub$maxfte) #check if these two variables are identical, res
ult in FALSE, so we can remove some dupes!

#only keep if certfte is equal to maxfte
dsub <- dsub %>% filter(certfte == maxfte) #now we are down to 46000 something.

anydupe(dsub) #there are now some of the cert that DO NOT contain dupes within school
year, but there are still some dupes. So we will move on.

dsub <- dupecol(dsub)
dsub <- narrow(dsub) #narrow down again
```

After applying the steps above, the data now has 46000 something rows. So I removed a good chunk of duplicates. I used the narrow() function to narrow the dataset down to the rows WITH duplicates; turn out there are only 7360 rows.

Starting with this dsub dataset with 7360 rows, I followed this step:

7. Using the variable assfte (assignment FTE), I removed rows that do not have maximum assfte.

```
#Looking at the data, the variable assfte is the assignment FTE.
#There are several assignment FTE within a year, so I will select only the biggest as
signment FTE per year

#first, create the max_assfte just like how I did it for certfte
dsub <- dsub %>% group_by(cert, schyear) %>% mutate(max_assfte = max(assfte))

#Only keep if assfte is equal to max_assfte
dsub <- dsub %>% filter(assfte == max_assfte) #we are now down to 6844 rows

#let's check by using our function
dsub <- dupecol(dsub)
anydupe(dsub) #there are STILL duplicates
table(dsub$dupyear) #we can see that 3310 rows contain dupes and 3534 rows do not con
tains dupes. Which is 50%:50% which is pretty good!
```

We are down to 3310 rows that contain duplicates. So I will narrow down even more and then apply these steps:

8. Using the variable asshpy (assignment hours per year), I remove the rows that contains assignment hours = 0.
9. I also remove the rows that asshpy is not equal to maximum assignment hour per year.

```
#let's narrow it down even more by selecting only the rows with dupes
dsub <- narrow(dsub)

#I can see that the variable asshpy (assignment hours per year) are 0 for many rows,
so I will remove those as well
dsub <- dsub %>% filter(asshpy != 0) #we are now down to 2974 rows
dsub <- dupecol(dsub)
table(dsub$dupyear) #we have 456 rows that do not have dupes - which is not a lot. So
we will need to do something more with the variable asshpy

#let's only keep one row per year that has the most asshpy - I am uncertain of this d
ecision but let's see how far it can go
dsub <- dsub %>% group_by(cert, schyear) %>% mutate(max_asshpy = max(asshpy))
dsub <- dsub %>% filter(asshpy == max_asshpy) #we are down to literally 390 rows.whic
h is little

#check for dupes
dsub <- dupecol(dsub)
anydupe(dsub)
table(dsub$dupyear)
```

We are now down to 1100 rows that contain duplicates.