Making Data Rodeo

Olivia Beck

2022-08-12

In this vignette, I will describe how I created the data-rodeo/dat-shinyapp.csv file that is used for the entire shiny app and all functions in it. Nothing on this document is evaluated, this is for purely explination purposes. In order to create the dat-shinyapp.csv, open the file making-data-wrangle/creating-dat-shinyapp.R and run the entire script.

Step 1: Set Up

First, load in the needed libraries and raw data.

```
library(dplyr)
library(readr)
library(stringr)

dat <- read_csv("data-raw/step-04-ceo-final.csv")
```

Next, get rid of any CEO's who are also CFO's, only select the columns we care about, and relabel them appropriately. We want to keep information regarding the EIN, filer name, filing year, total expenses, total employees, total assets, gross receipts, NTEE codes, university/hospital status, transition year status, CEO total compensation, and CEO gender. From this information, we create the variable coding we want to use (i.e. MajorGroup, NTEE, NTEE.CC).

```
dat.sec <- dat %>%
  #get rid of anybody who is also a CFO
  filter(CFO == F) %>%
  #Select only the columns we care about
  select(EIN, FilerName1,
    NTEE1, NTEEFINAL, TOTALEXPCURRENT, TOTEMPLOYEE, UNIV, HOSP,
    TOTCOMP, #TOTCOMP is totalComp adjusted to 2020 inflation rate
    gender, FormYr, STATE, FORMTYPE,
    GROSSRECEIPTS,TOTALASSETSENDYEAR,
    transitions, TRANS.D) %>%
  #Creating Major Group
  rename(NTEE = NTEE1) %>%
  rename(NTEE.CC = NTEEFINAL)%>%
  mutate(MajorGroup = case_when(NTEE %in% LETTERS[1] ~ 1,
                                NTEE %in% LETTERS[2] ~ 2,
                                NTEE %in% LETTERS[3:4] ~ 3,
                                NTEE %in% LETTERS[5:8] ~ 4,
                                NTEE %in% LETTERS[9:16] ~ 5,
                                NTEE %in% LETTERS[17] ~ 6,
```

```
                                NTEE %in% LETTERS[18:23] ~ 7,
                                NTEE %in% LETTERS[24] ~ 8,
                                NTEE %in% LETTERS[25] ~ 9,
                                NTEE %in% LETTERS[26] ~ 10)) %>%
  #Renaming columns
  rename(Name = FilerName1) %>%
  rename(TotalExpense = TOTALEXPCURRENT) %>%
  rename(TotalEmployee = TOTEMPLOYEE) %>%
  rename(TransitionYr = transitions) %>%
  rename(CEOCompensation = TOTCOMP) %>%
  rename(GrossReceipts = GROSSRECEIPTS) %>%
  rename(TotalAssests = TOTALASSETSENDYEAR) %>%
  rename(Gender = gender) %>%
  rename(State = STATE) %>%
  rename(FormType = FORMTYPE)
```

Reassign any org with less than 0 total expenses to just have 0 total expenses. Do the same for total assests.

```
dat.sec <- dat.sec %>%
  #if TotalExpense<0, assign TotalExpense=0
  mutate(TotalExpense = case_when(TotalExpense >= 0 ~ TotalExpense ,
                                  TotalExpense < 0  ~ 0))%>%
  #if TotalAssests<0, assign TotalAssests=0
  mutate(TotalAssests = case_when(TotalAssests >= 0 ~ TotalAssests ,
                                  TotalAssests < 0  ~ 0))
```

Adjust for inflation. The raw data file has all CEO compensation in 2020 dollars. The 2022 inflation has been skyrocketing. We just used January 2022 as the base line for inflation rate, then whenever this project gets published, the inflation rate can easily be changed.

```
dat.sec <- dat.sec %>%
  #Adjust for inflation , *1.09
  mutate(CEOCompensation = CEOCompensation * rate.2020.to.2022)
```

For each unique organization, only keep the most recent entry. Note, this causes issues for the gender pay gap graph for graphing over time. These issues are discussed in the gender-pay-graph vignette.

```
dat.sec <- dat.sec %>%
  #Only keep the most recent Year for each unique EIN
  group_by(EIN) %>%
  filter(FormYr==max(FormYr)) %>%
  ungroup()%>%
  #keep distinct entires, there are some entries in original dat that are multiple.
  distinct()
```

Change all org names to all caps for consistance. Reorder the table to be more logical.

```
dat.sec <- dat.sec %>%
  #make org names all caps
  mutate(Name = toupper(Name)) %>%
  #Reorder columns
  relocate(FormYr, FormType, Name, EIN, State, MajorGroup, NTEE, NTEE.CC, UNIV, HOSP,
```

```
            TotalExpense, TotalEmployee, GrossReceipts,TotalAssests,
            CEOCompensation, Gender,
            TransitionYr)
```

Step 2: Adding location types

We will be adding location types by FIPS codes and RUCA codes. The BMF files have the FIPS codes for each EIN. We then match FIPS codes to RUCA codes for type of city. We then aggragate RUCA codes into "Metropolitan" or "Rural".

More information about FIPS codes can be found here:

More information about RUCA codes can be found here: https://www.ers.usda.gov/data-products/rural-urban-commuting-area-codes/documentation/

First, get all unique EIN's

```
#Our EIN's
ein.ours  <- dat.sec$EIN
```

Getting FIPS codes Get the link for the BMF flies. Some years have multiple files. We used the most recent file for each year.

```
links <- c("https://nccs-data.urban.org/dl.php?f=bmf/2019/bmf.bm1908.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2018/bmf.bm1812.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2017/bmf.bm1712.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2016/bmf.bm1608.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2015/bmf.bm1512.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2014/bmf.bm1412.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2013/bmf.bm1312.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2012/bmf.bm1212.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2011/bmf.bm1112.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2010/bmf.bm1011.csv",
           "https://nccs-data.urban.org/dl.php?f=bmf/2009/bmf.bm0910.csv"
           )
```

We will start with the first link, but then for the remainder of the link do the exact same process in a for loop. The process is, read in the link, only keep the EIN, zip code, and FIPS code, only keep the EIN's that are in our dat.sec, then store those EIN/Zip/FIPS codes in a data file to later merge into dat.sec. Note it can take up to 10 minutes to load in each BMF file.

```
#start with the first link
bmf.all <- read_csv(links[1])
bmf.all <- bmf.all[ , c("EIN", "ZIP5", "FIPS")]

#list of total EIN's
ein.total <- bmf.all$EIN

#list of rows to keep
keep.rows <- ein.total %in% ein.ours

#data set to merge
bmf.all <- bmf.all[keep.rows, ]
```

Now do the exact same things for the other links, except only keep the EIN's thare aren't in dat.sec AND not in the previous bmf files.

```
#now do the other links
counter = 1
for(i in 2:length(links)){
  #upload new one
  bmf.i <- read_csv(links[i])
  bmf.i <- bmf.i[ , c("EIN", "ZIP5", "FIPS")]

  #get all EIN's
  ein.all <- bmf.all$EIN

  #get new EIN's
  ein.new <- bmf.i$EIN

  #keep new ones
  keep.rows <- (! (ein.new %in% ein.all)) & (ein.new %in% ein.ours)

  #add to data frame
  if(sum(keep.rows >0 )){  bmf.all <- rbind(bmf.all, bmf.i[keep.rows, ])}

  counter <- counter +1
}
```

Now merge the dat.bmf and dat.sec by EIN. Also, deal with the leading 0 issue by making FIPS codes a character vector and manually adding back in the leading 0.

```
#merge bmf.all and dat.sec by EIN and ensure all FIPS codes have 5 digits
dat.with.geo <- merge(dat.sec , bmf.all, by = "EIN") %>%
  #99 of them dont have FIPS #'s
  #filter these out for now. ask Jesse how to deal with these later
  filter(!is.na(FIPS)) %>%
  #make as character without a decimal
  mutate(FIPS = as.character(round(as.numeric(FIPS)))) %>%
  #get number of digits
  mutate(digit =  stringr::str_length(FIPS)) %>%
  #fix the ones with 4 digits to be 5 digits (leading 0 problem)
  mutate(FIPS = case_when(digit == 4 ~ paste0("0", FIPS),
                          digit == 5 ~ FIPS))
```

Getting RUCA codes The USDA provides all RUCA codes for every FIPS code. We assign every EIN its RUCA code through the FIPS code.

Load in the FIPS to RUCA codes and format the data so that it can be merged.

```
#get RUCA codes from FIPS codes
dat.ruca.raw <- readxl::read_excel("data-raw/cencus-ruca-2010-revised.xlsx",
                                   skip = 1)

#format so it can be merged
dat.ruca <- dat.ruca.raw %>%
  #select only the columns we care about
```

```
  select("State-County FIPS Code",  "Primary RUCA Code 2010" )%>%
  #rename them to something readable
  rename( FIPS = "State-County FIPS Code", #state-county code
          RUCA = "Primary RUCA Code 2010") %>%
  #get only the ones in our data set
  filter((FIPS %in% unique(dat.with.geo$FIPS))) %>%
  #group by FIPS.SC and take the avearge RUCA number
  group_by(FIPS) %>%
  summarise(RUCA = mean(RUCA), .groups = "keep")
```

Merge dat.ruca and dat.with.geo. Assign any EIN with a RUCA code of 6 or lower to be "Metropolitan" and a RUCA code of 7 or higher to be "Rural".

```
#merge
dat.merge <- merge(dat.with.geo, dat.ruca, by = "FIPS") %>%
  #make RUCA into rural and metropolitan
  mutate(LocationType = case_when(RUCA <= 6 ~ "Metropolitan",
                                  RUCA > 7 ~ "Rural")) %>%
  #get rid of intermediate steps
  select(-c(RUCA, digit)) %>%
  relocate(FIPS, .after = LocationType)
```

Imputing total employees for 990EZ filers

Form 900EZ does not have any recorded information about total employees. We use information from the full 990 filers who qualify to file the 990EZ but for whatever reason did not, to impute the total employees for the 990EZ filers.

Split the data by who filed a 990 and who filed a 990EZ.

```
#make 2 data sets for 990 and 990EZ
dat.990 <- dat.merge %>%
  filter(FormType == "990")

dat.EZ <- dat.merge %>%
  filter(FormType == "990EZ")
```

Find the orgs who filed a full 990 but qualified to file a 990EZ.

```
#filter 990 data to match criteria for 990EZ filers
dat.model <- dat.990 %>%
  filter(GrossReceipts < 200000) %>%
  filter(TotalAssests < 500000)
```

Fit a regression model with Total Employees as the response variable.

```
dat.model$FormYr <- as.factor(dat.model$FormYr )
dat.model$MajorGroup <- as.factor(dat.model$MajorGroup)
dat.model$NTEE <- as.factor(dat.model$NTEE )
dat.model$UNIV <- as.factor(dat.model$UNIV )
dat.model$HOSP <- as.factor(dat.model$HOSP )
dat.model$State <- as.factor(dat.model$State)
```

```
#remove non-complete cases
dat.model <- na.omit(dat.model)

mod <- glm(TotalEmployee ~ FormYr + MajorGroup + UNIV +  HOSP +
             TotalAssests + GrossReceipts  + TotalExpense ,
           family="poisson", data = dat.model )
#checking that model is okay
# dat.plot <- data.frame(dat.model$TotalEmployee, mod$fitted.values) %>%
#   filter(dat.model$TotalEmployee < 15)
#
# plot(dat.plot[, 1], dat.plot[, 2],
#      xlab = "observed",
#      ylab = "fitted")
# abline(0,1, col = 2)
#
# plot(mod)
```

Impute the 900EZ filers total employees. Also, force any imputed value less than 0 to just be assigned 0.

```
## Get the imputed data
#format EZ data
dat.EZ$FormYr <- as.factor(dat.EZ$FormYr )
dat.EZ$MajorGroup <- as.factor(dat.EZ$MajorGroup)
dat.EZ$NTEE <- as.factor(dat.EZ$NTEE )
dat.EZ$UNIV <- as.factor(dat.EZ$UNIV )
dat.EZ$HOSP <- as.factor(dat.EZ$HOSP )
dat.EZ$State <- as.factor(dat.EZ$State)

#impute
dat.EZ$TotalEmployee <- stats::predict.glm(mod, dat.EZ)

#if anything is less than 0 just make it 0
dat.EZ$TotalEmployee <- ifelse(dat.EZ$TotalEmployee > 0 , dat.EZ$TotalEmployee, 0)
```

Finally, combine the 990 and 990EZ filers into one data set.

```
## Combine the two data sets
dat.final <- rbind(dat.990, dat.EZ) %>%
  distinct()
```

Hard coding out problematic cases

There are a few cases that were causing erros in the HEOM function. I think its because somehow their NA values got coded as the character "<NA>" which the HEOM function doesn't know how to process.

```
### Remove these rows. They have no useful data because too many NA's
#plus they are causing issues in the HEOM function
#13072 13075 13083 13103 13106 13111 13114 13117 13127 13139 13142

get.rid.of <- which(is.na(dat.final$MajorGroup) & is.na(dat.final$NTEE) & is.na(dat.final$NTEE.CC))
get.rid.of <- c(get.rid.of, 13139, 13142)
```

```
dat.final <- dat.final[ -get.rid.of, ]
```

Save the file

Save this data set as data-rodeo/dat-shinyapp.csv

```
write_csv(dat.final, file = "data-rodeo/dat-shinyapp.csv")
```