Written report

ChunEn Chu

2021/12/12

Abstract Public Policy Research Institute(PPRI) serves as a leading interdisciplinary government and social policy research organization. It is also our project’s client. In this project, we will use the given data from PPRI to design and build a long term of code to automate process the data into banners and top-lines for their website. To rich the goal of the project it will requires several package such as kable, knitr and labelled. we will illustrate the code and the results. At the end, our final goal of this project is to create a package that anyone researchers can apply it into their system and use them to produce banners and top lines from any kind of data sets.

Introduction The data sets been used in this project will be explain through this report. The project goal also has a sample from our client. The goal of out project is to create the same banner and top line by using R code. After introducing the data, we will discuss about the sample. Then, we will display the code and explain how it works. The first part of the code will show the way to produce a part of our sample. The next step is to create another table. Finally, the table will be combine together. After create the correct code to produce the banner. we will display the idea to create the function that can apply any data set and transform into a banner. Lastly, the function will be apply with for loop function which is our project’s ultimate goal automate process survey data into banner and top lines. All of the function and code will be convert into a package with full of comment.

Data & Sample From the Fig.1 we can see that our data has 5851 records, 198 columns inside the data and include huge information. we will have to design a code that can select several column we need. After selecting the column, we will produce separate table and combine the table to get the similar table from the sample we have as Fig.2.

Code & Method In this part, we will include the important part of code and explain from this project.

library(tidyverse)  
library(foreign)  
library(labelled)  
library(haven)  
library(readstata13)  
library(pollster)  
library(knitr)  
library(kableExtra)  
library(sjlabelled)  
library(rlang)

The code above are the package we will use and work through the project to create a complete banner for PPRI. They are several important package inside the code. The “labelled” package allows people to use get\_label and labels series function which is helpful since the data has more than one layers of labels in some column. The “haven” and “pollster”package can produce cross table from out data sets. “readstata13” package can read sta. file for our R system. The “kableExtra” package can let r studio customize the kabel we made from cross table. For “sjlabelled”, it can replace and recode the labels inside the data. And for “rlang” package, we will use it to solve the problem in for loop function.

dta = read\_dta("AVA\_PRRI\_Wave 2\_final\_wtd\_banners\_v3.dta")  
sav = read\_sav("AVA\_PRRI\_Wave 2\_final\_wtd.sav")

The code read\_dta and read\_sav, import the dta and save types file into our system

count(sav)

## # A tibble: 1 x 1  
## n  
## <int>  
## 1 5851

Compare with the sample we get and make sure we have the correct sample size as the sample. After that we will use view() function to check our the column related to age and education. From the result we can see that the column are in dta data sets.

dta%>%  
 select(Respid,age, educ) ->dta1  
sav%>%  
 merge(dta1, by = "Respid") -> test1

In this part, we select Respid, age, educ column from dta data sets, by doing this we can avoid duplicate problem when we combine two data sets into one. The column “Respid” is the key to combine two data sets. The merge function can help us combine two data sets by using the key column.

#get\_labels(test1$Q2\_1)  
#get\_labels(test1$educ)

By using this function we can check out the labels inside the column.

#unique(test1$Q2\_1)  
#unique(test1$educ)

The function shows the unique attributes inside question and education column

test1 %>%  
 select(age)%>%  
 transmute(group = ifelse(age %in% 18:29, "B",  
 ifelse(age %in% 30:49, "C", ifelse(age %in% 50:64,   
 "D", ifelse(age %in% 65:150, "E", "skip"))))) ->t1  
test1 <- cbind(test1, as.vector(t1))

In this part of code, we select the age column in our data and use transmute to create a new column by a new rule. we separate the age by different group such as 18-29 and give it a letter “B” and assign it to t1. Then we use cbind function which can combine column together to add our age group column to test1 data.

#test1$educ

The code will show the groups inside the data. Compare to the sample we got the original data has 6 group for education but in the sample only has 5 groups.

values <- c("High school or Less", "High school or Less", "Some College", "College Graduate", "Post-Graduate", "skip")  
test1$Education <- values[test1$educ]  
test1%>%  
 select(Education, educ)  
test1%>%  
 mutate(edu\_group = recode(Education, "High school or Less" = "F",  
 "Some College" = "G",  
 "College Graduate" = "H",  
 "Post-Graduate" = "J")) ->test1

In order to convert two of the group into high school or less. we create a vector called values. After that we can use base R code to apply the vector and add a new column called education. Then, we use mutate function to edit the education colunm give them a group letter like what we have in the sample banner.

test1%>%  
 crosstab(x= Q2\_1, y = group, weight = weight,pct\_type = "column") ->t1  
  
test1%>%  
 crosstab(x= Q2\_1, y= edu\_group, weight = weight, pct\_type = "column") ->t2  
t1

## # A tibble: 6 x 5  
## Q2\_1 B C D E  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 Skipped 0.720 0.353 0.569 0.727  
## 2 Very worried 15.2 18.4 12.7 8.76   
## 3 Somewhat worried 31.3 34.2 29.2 26.2   
## 4 Not too worried 30.9 28.8 38.3 39.4   
## 5 Not at all worried 22.0 18.3 19.3 24.9   
## 6 n 1091. 1894. 1619. 1248.

t2

## # A tibble: 6 x 5  
## Q2\_1 F G H J  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 Skipped 0.927 0.414 0.306 0.219  
## 2 Very worried 19.3 11.8 10.7 9.98   
## 3 Somewhat worried 29.8 28.6 33.9 32.4   
## 4 Not too worried 30.6 35.2 36.3 38.0   
## 5 Not at all worried 19.4 24.0 18.8 19.3   
## 6 n 2210. 1766. 1038. 837.

After making the groups for age and education column. Produce separate table for our banner, apply the data weight and display by percentage

merge(t1, t2, key = t1$Q2)-> t3  
All\_Americans\_A <- apply(t3%>%select(-Q2\_1),1,sum)/8  
  
All\_Americans\_A[1] <- All\_Americans\_A[1]\*4  
  
t4 <- cbind(All\_Americans\_A,t3)  
#t4$Q2\_1[1] <- "Unweighted sample size"  
t4[t4 =="n"] <- "Unweighted sample size"  
t4

## All\_Americans\_A Q2\_1 B C D  
## 1 5850.9969000 Unweighted sample size 1090.615800 1893.5812000 1619.0564000  
## 2 20.7446378 Not at all worried 21.961070 18.2508255 19.2906189  
## 3 34.6778732 Not too worried 30.856091 28.7795422 38.2826133  
## 4 0.5292829 Skipped 0.719896 0.3529978 0.5686893  
## 5 30.7008196 Somewhat worried 31.257424 34.2421334 29.1678289  
## 6 13.3473865 Very worried 15.205520 18.3745012 12.6902497  
## E F G H J  
## 1 1247.7435000 2210.0529000 1766.2068000 1037.5621000 837.1751000  
## 2 24.9478519 19.3618714 23.9878252 18.8100356 19.3470040  
## 3 39.3907402 30.5885936 35.2097218 36.3022030 38.0134813  
## 4 0.7265516 0.9267018 0.4143286 0.3064106 0.2186878  
## 5 26.1716451 29.8421861 28.6372355 33.8500992 32.4380049  
## 6 8.7632113 19.2806471 11.7508890 10.7312517 9.9828220

Combine age and education table into one table and ues apply function to caculate the average of each row and create new column called all american. Then add the correct unweighted sample size number to all american’s column and add it to the first table we build. And replace the row n into unweight sample size with two options.

#t4%>%  
# relocate(All\_Americans\_A, .after = Q2\_1)%>%  
 #rename(" " = "Q2\_1")%>%  
# kable(caption ="Q2.a How worried, if at all, are you about each of the following? You or someone in your family will get sick from the coronavirus", digits = 0,   
# col.names = c(" ", "All\_Americans\nA", '18-30\nB', '30-49\nC', '50-64\nD', '65+\nE',  
# 'HighSchool or Less\nF', 'Some College\nG', 'College Gradute\nH',  
# 'Post-Gradute\nI')) %>%  
# kable\_styling("striped", full\_width = F,  
# position = "left", font\_size = 12) %>%  
# add\_header\_above(c(" " = 1, "All Americans" = 1, "Age groups" = 4, "Education" = 4))

After combine the whole table, now we have to edit our table. First, we use the relocate function to add all american’s column to a right location which is after question column. Then, import the table into kable() function so that we could customize the formatting and add thing inside the table. By using caption = we can add our question for this table. The col.names help us rename our column inside the table but we have to make sure give all column a name otherwise, it will shows an error for column not match. Since we do not nedd the first column’s name we can give a space for it and the function will remove the column name for first column. And back slash “n” mean a space in R code. For kable\_styling(), we can adjust our print width, font size and other format style in this function. The add\_header\_above function helps to create a subtitle that can group our table into several groups. And the digit = 0 means remove all decimal numbers inside the table.

create\_crosstab <- function(x) {  
 x %>%  
 select(age)%>%  
 transmute(group = ifelse(age %in% 18:29, "B",  
 ifelse(age %in% 30:49, "C", ifelse(age %in% 50:64,   
 "D", ifelse(age %in% 65:150, "E", "skip"))))) ->t1  
x <- cbind(x, as.vector(t1))  
values <- c("High school or Less", "High school or Less", "Some College",  
 "College Graduate", "Post-Graduate", "skip")  
x$Education <- values[x$educ]  
x%>%  
 mutate(edu\_group = recode(Education, "High school or Less" = "F",  
 "Some College" = "G",  
 "College Graduate" = "H",  
 "Post-Graduate" = "I")) ->x  
crosstab(df = x, x= Q2\_1, y = group, weight = weight,pct\_type = "column") ->t1  
  
crosstab(df = x, x= Q2\_1, y= edu\_group, weight = weight, pct\_type = "column") ->t2  
merge(t1, t2, key = t1[1])-> t3  
All\_Americans\_A <- apply(t3%>%select(-Q2\_1),1,sum)/8  
  
All\_Americans\_A[1] <- All\_Americans\_A[1]\*4  
  
t4 <- cbind(All\_Americans\_A,t3)  
t4$Q2\_1[1] <- "Unweighted sample size"  
t4%>%  
 relocate(All\_Americans\_A, .after = Q2\_1)%>%  
 #rename(" " = "Q2\_1")%>%  
 kable(caption =attr(x$Q2\_1, 'label'), digits = 0,   
 col.names = c(" ", "All\_Americans\nA", '18-30\nB', '30-49\nC', '50-64\nD', '65+\nE',  
 'HighSchool or Less\nF', 'Some College\nG', 'College Gradute\nH',  
 'Post-Gradute\nI')) %>%  
 kable\_styling("striped", full\_width = F,  
 position = "left", font\_size = 12) %>%  
 add\_header\_above(c(" " = 1, "All Americans" = 1, "Age groups" = 4, "Education" = 4))  
}

Apply the code we used before into function and replace the data import to x.

#colnames(test1)  
#colnames(test1)%>%  
# str\_detect("^Q")  
#str\_extract(colnames(test1), "^Q\\d\_\\d|^Q\\d\\w|\_\\d|Q\\d\\d|^Q\\d\\w\\d\_\\w")

After build the function, we have to apply multiple question column into the function. The colnames() function display the column names of each column in our data sets. And it is obvious that the question column is start with an capitol letter “Q”. By using str series function in R. We can easily find out the question columns in the data.

cc <- test1%>%  
 colnames()%>%  
 str\_subset("^Q")

Create a vector called cc that include all question column’s name.

#for (i in cc) {  
# table1 <- crosstab(df = test1, x = i, y = group, weight = weight,pct\_type = "column")  
# print(table1)  
#}  
#library(rlang)  
#for (i in cc) {  
# table2 <- crosstab(df = test1, x = !!sym(i), y = group, weight = weight,pct\_type = "column")  
# print(table2)  
#}  
#library(rlang)  
#col1 <- colnames(test1)  
#for (i in cc) {  
# table1 <- crosstab(df = test1, x = !!sym(i), y = group, weight = weight,pct\_type = "column")  
# print(table1)  
#}

Then, write a for loop function that apply the all column inside the cc column. The first for loop function failed since the system can not return the list in side i column so we have to use !!sym() function that from “rlang” package that can return the list inside the column then we can get our correct table from the for loop function.

age\_crosstab1 <- function(x) {  
 x %>%  
 select(age)%>%  
 transmute(group1 = ifelse(age %in% 18:29, "B",  
 ifelse(age %in% 30:49, "C", ifelse(age %in% 50:64,   
 "D", ifelse(age %in% 65:150, "E", "skip"))))) ->t1  
x <- cbind(x, as.vector(t1))  
values <- c("High school or Less", "High school or Less", "Some College",  
 "College Graduate", "Post-Graduate", "skip")  
x$Education <- values[x$educ]  
x%>%  
 mutate(edu\_group1 = recode(Education, "High school or Less" = "F",  
 "Some College" = "G",  
 "College Graduate" = "H",  
 "Post-Graduate" = "I")) ->x  
for (i in str\_subset(colnames(x),"^Q")[1:5]) {  
 crosstab(df = x, x= !!sym(i), y = group1, weight = weight,pct\_type = "column") ->t1  
 crosstab(df = x, x= !!sym(i), y= edu\_group1, weight = weight, pct\_type = "column") ->t2  
merge(t1, t2, key = t1[1])-> t3  
All\_Americans\_A <- apply(t3%>%select(-i),1,sum)/8  
  
All\_Americans\_A[1] <- All\_Americans\_A[1]\*4  
  
t4 <- cbind(All\_Americans\_A,t3)  
t4[t4 =="n"] <- "Unweighted sample size"  
#t4%>%  
# as\_tibble()-> t4  
#t4 <- t4[c(6,2,3,4,5,1),]  
t4%>%  
 relocate(All\_Americans\_A, .after = i)%>%  
 kable(caption =get\_label(x[i]), digits = 0,   
 col.names = c(" ", "All\_Americans\nA", '18-30\nB', '30-49\nC', '50-64\nD', '65+\nE',  
 'HighSchool or Less\nF', 'Some College\nG', 'College Gradute\nH',  
 'Post-Gradute\nI')) %>%  
 kable\_styling("striped", full\_width = T,  
 position = "left", font\_size = 8) %>%  
 add\_header\_above(c(" " = 1, "All Americans" = 1, "Age groups" = 4, "Education" = 4))%>%  
 print()  
}  
}

This is one of the final banner function we will get from the code after apply everything inside the for loop function.

#dta%>%  
# colnames()%>%  
# str\_subset("edu|EDU") ->e1  
#sav%>%  
# colnames()%>%  
# str\_subset("edu|EDU")->e2  
#dta%>%  
# select(e1)%>%  
# get\_label()

Using the str function to find the actual column we need.

#test2%>%  
# group\_by(race\_m)%>%  
# count()%>%  
# table()%>%  
# colnames()%>%  
# as.numeric()%>%  
# rev()->size1  
#size1  
#a <- c(1,size1)  
#r1%>%  
# filter(Q2\_1 != "n") %>%  
# rbind(a) ->r1

calculate the underweighted sample size for race\_m column and assign it to size1. Add the unweighted sample size back to the original table r1.

## Warning in recode.haven\_labelled(polaffil, `1` = "Republican", `2` =  
## "Democrat", : The type of .x has been changed and value labels attributes have  
## been lost.

## # A tibble: 5,086 x 1  
## polaffil1   
## <chr>   
## 1 Independent   
## 2 Republican   
## 3 Independent   
## 4 Independent   
## 5 Independent   
## 6 Independent   
## 7 Something else  
## 8 Independent   
## 9 Independent   
## 10 Democrat   
## # ... with 5,076 more rows

## label 'Strong Republican' was replaced with new value label.  
## label 'Not very strong Republican' was replaced with new value label.  
## label 'Lean Republican' was replaced with new value label.  
## label 'Independent' was replaced with new value label.  
## label 'Lean Democrat' was replaced with new value label.  
## label 'Not very strong Democrat' was replaced with new value label.  
## label 'Strong Democrat' was replaced with new value label.

These are several ways to clean the data. we can use filter function to remove the values we do not need. The other way is to rename the values in the column and then use na.omit() function to remove the NA or we can also use replace\_labels() function to change the labels inside the data.

#r5%>%  
# as\_tibble()-> r5  
#it is necessary to use as\_tibble() function otherwise we will create a non exsist column for the re-order  
#r5 <- r5[c(6,2,3,4,5,1),]

The code above is a way to reorder our row from the table we have. It is important that we have to use as\_tibble function to avoid creating invisible column to our system. This will be a problem when we make a cross table.

For other type of banners, there are some different at the begin. The race\_region table do not need to create group for the column but we have to calculate the unweighted and apply into the table. The politics and religious table there are some unnecessary columns exist we can use base R code to remove the column. Adding space to the column names of our table can avoid the text stack together.

Overall, the technique to produce the banner from our data is to deconstruct the table to several parts. And produce it with code. Finally, combine every code into a for loop function.

Conclusion In the past few months, the function to automate process survey data has come with a prototype and it is able to print different type of table for banner from out data sets. There are many challenges during the progress such as using the proper code to get the correct result. Studying more about the data we have. By the end, when we produce the table step by step. We have the proper code that can automate process survey data. And the code has been simply and improve every weeks. The code now are able to print every table for different type of banner. After, the code has been complete, our team build a package for our function and also include the top-line function. All of the function inside the package have comment and explanation for the people in the future.

Future step The function for banner still can be improve. First, the function may still have some space to be simplify or shorter. The comments for the script file still need improve. Also, the banner function are still separate by different type of banner we need. From the other sample data we got from our client that a function to detect the required column will be helpful to our for loop function so the we can combine all functions into one function. The formatting of the banner are still not perfect yet. There are still many place that we can improve