**Epi 510, R cumulative assignment**

For this assignment, please submit two files: (1) **a file containing text, tables and figures** (.doc, .docx, .pdf or .txt) responding to questions posed in the assignment, and (2) **an R script** that performs the requested operations (.R).

You’ll find the BRFSS dataset and documentation on the course Canvas site. We’ll use this dataset for the survey methods sessions in this course, and you’ll use it more extensively in EPI 514. To properly analyze this dataset, you’ll need to account for survey weights. Since our purpose here is primarily to practice coding and applied analysis skills, and since you’ll likely want to start on this assignment before we cover survey weights, we’ll ignore the weights for most of this assignment. The last set of questions will require you to account for those weights. That way you can work on most of the assignment on your own timeframe, but still have the opportunity to practice the material on survey weights. You’ll need two files: the BRFSS dataset (LLCP2017.XPT), and a file that will link FIPS codes to state names and abbreviations (fipsLink.csv).

One final note of introduction: this assignment will walk you through a complete analysis from importing, managing, and cleaning data, to exploration and descriptive analyses, to graphs and inferential analyses. The raisons d'être for this assignment are to be a refresher of what you've done in class and other assignments and, more importantly, to tie everything together so you come away with a cohesive picture of the process and flow of a real analysis. With that last goal in mind, this assignment will have more macro-scale explanatory narrative than most.

1. Let’s start our analysis as per usual: start up RStudio and open up a new script.  
   1. At the top of the script write a comment block to indicate what the script will do, it’s dependencies (i.e. any data files, libraries, or other .R files that the script will need to run), who wrote it, when it was written, and any other information you think would be useful to future users of the script. You might find it easiest to start with basic information and add information about the purpose and function of the script later when you have a clearer sense of this yourself. Just don’t forget to come back and do this! **(3 points)**
   2. Include code to clear the R session, set your working directory. **(2 points)**
2. We’re going to start with the LLCP2017.XPT file. This file is in SAS XPORT format, and it’s very large, so let’s tame this beast.   
   1. Load the “tidyverse” and “haven” libraries. **(2 points)**
   2. Use read\_xpt to import the SAS transport format BRFSS data file. **(2 points)**
   3. Haven's read functions don't coerce variables to the most efficient storage format and, when variables have names that are disallowed in R, they rename them awkwardly. We can fix both issues by saving the data to a .csv file and then reading this .csv back in. Use write.csv to save the BRFSS dataset to a .csv file. Then use read.csv to read this file back in. **(4 points)**
3. Our BRFSS dataset indicates the state in which each interview was conducted by its FIPS (Federal Information Processing Standard) code. Since most of us don’t know these codes off the tops of our heads, we’re going to want connect each FIPS code to the corresponding state name. The fipsLink file contains the name, abbreviation, and FIPS code for each US state and territory.   
   1. Read in the fipsLink.csv file. **(2 points)**
   2. The variable containing FIPS codes is called “X\_STATE” in the BRFSS dataset, and called "fips" in the fipsLink file. We’ll want to merge the BRFSS and fipsLink data on this variable. Use the merge function with by.x and by.y to merge on these variables; we'll only want to keep observations that were in the BRFSS dataset (i.e. we don't need the fips codes for locations that aren't in the BRFSS dataset), so choose the appropriate option (i.e. choose the appropriate option among all, all.x, and all.y) . **(3 points)**
   3. R doesn't produce a merge report but we can still determine if all of the rows of the BRFSS dataset successfully matched the fipsLink dataset. Choose a variable that was in the fipsLink dataset (and had no missing values in that dataset) and determine if there are any missing values of that variable in the new merged dataset. If you have no missing values then all rows merged completely. Did all rows match up properly? **(2 points)**
4. Now that you have the pieces together, you’ll want to take a look at the dataset. We’ll inspect things more carefully once we’ve pruned the dataset to only variables of interest. For now, use either glimpse (or str) and View, just to get a rough sense of things. **(2 points)**
5. The full data set is quite large (358 variables and >450k observations). Once you know which variables and observations you need for your analysis it’s good practice to drop everything else to reduce your working file size, and avoid unnecessary clutter (and the potential for clutter-induced paralysis). As long as you’ve retained the original data, there’s no reason to fear dropping data: you can always reload the original data if you realize that you need a variable you dropped. For this assignment we’ll keep thirteen variables: "name", "postalcode", "X\_STATE", "X\_PSU", "X\_STSTR", "X\_LLCPWT", "SEX", "X\_AGEG5YR", "X\_AGE65YR", "PHYSHLTH". Go ahead and drop all other variables. **(3 points)**
6. Now that you have a manageable number of variables, you’ll want to look at your dataset again, a bit more carefully than before.
   1. Use glimpse and View as before. **(2 points)**
   2. Now that we have fewer variables, you can also run summary on the entire data frame. Look at the variables for bad formats and impossible or unlikely values. Do any variables appear to be coded in the wrong format (e.g. numeric variables stored as characters)? Are there values you see (either min or max) that seem likely to indicate some form of missingness? What is an example of one such value? **(3 points)**

1. We’ve looked at the big picture, now it’s time to dig into each variable one-by-one, check the coding, and clean things up. You’ll want to have the data dictionary handy for this, since you’ll need to determine how each variable is coded.
   1. Rename all variables to names that are both reasonably descriptive and typeable. **(5 points)**
   2. Convert all values corresponding to “Don’t know/Not sure” and “Refused” to NA for physhlth . Look at the data dictionary carefully on this one: there is a code that you could easily mistake for missing, but should be converted to a number of days! Now create two binary versions of this variable: one version that is a labelled factor variable coded as 1 = 15 or more days, and 2 = <15 days; and another logical version of each using 0/1 coding (0 = <15 days and 1 = 15+ days). **(3 points)**
   3. For the variable that contains five-year age groups (\_ageg5yr), convert all rows with “Don’t know/Refused/Missing” values to NA, and convert this to a labelled factor variable. Tabulate the final labelled variable to make sure that things look right. **(3 points)**
   4. Create two versions of the binary variables \_age65yr and sex: for each one create a version that is a labelled factor variable coded as 1=65 or older/male, 2=18 to 64/female; and create one that is a logical variable coded as 0=False, 1=True (name this variable either “male” or “female” for sex, and code it accordingly). For all variables, convert all rows with “Don’t know/Refused/Missing” values to missing. **(4 points)**
   5. Now that you have your variables created and cleaned, you'll want to put them in a sensible order. Choose an order that makes sense to you. I recommend grouping related variables together (e.g. all of the location variables, followed by all of the date variables, followed by the demographic variables, etc.). **(2 points)**
   6. At this point you should have a clean dataset, and you’ll probably not want to have to run all of the data management code repeatedly. Save the clean dataset as “brfss2017Clean.rds”, save your script with a name that will let you (and others) know that it is the data management script (I’ll use “rCumulative\_01\_dataMngmt.R”). **(2 points)**
2. Now that we have a clean dataset, we're going to move on to more formal data exploration and descriptive analysis. This is an essential part of every analysis that is often done in excess haste and with an insufficiently critical eye. Take the time here to really understand your data and you'll face far fewer mistakes and unpleasant surprises in the later phases of your analysis.  
   1. Create a new R script. As before, include a comment header, and the typical code needed to setup your R session (rm(list = ls()), loading libraries, etc.). **(3 points)**
   2. Read in the clean rds version of the BRFSS dataset that you created in the previous question. **(2 points)**
   3. Apply the summary function to the full dataset Look through the output for each variable to ensure that everything looks correct. If there's a problem, go back to the data management script, fix the problem and rerun it. If everything looks good, then you'll want to look at the tables to get a sense of your data. I like to treat this step like an interrogation in which I ask questions of my data: How many observations do you have? What is the breakdown by sex? How much missingness is there? Is the missingness so great for any variable that I have to question its usefulness? (you don’t need to provide a written answer here) **(3 points)**
   4. Create histograms for the continuous poor physical health day variable, and for the age group variable. You may use either the base R hist function or ggplot. Choose options that make sense for each variable and produce histograms that you'd find useful as an analyst. As we go forward and need to make decisions about our analysis, we'll need to understand the nature of the distribution of these continuous variables as it relates to assumptions of different modelling approaches (e.g. linear regression assumes a normally distributed continuous outcome). For each variable describe the distribution in a few words (e.g. normal distribution, uniform distribution, right-skewed, left-skewed, multi-modal, etc.). **Paste your histograms into the document with your submission.**  **(6 points)**
   5. We’d like to know if the reported numbers of poor physical health days vary by age.   
      1. Use either dplyr functions or tapply to create a new data.frame that contains the mean of the poor physical health variable (continuous version) by age group. **(4 points)**
      2. Use ggplot to create a connected scatter plot (i.e. overlay a geom\_point and geom\_line) to visualize the age pattern of the mean variable. **(2 points)**
3. Now that you’ve done exploratory and descriptive analyses, you’ll want to move on to looking at associations. We want to understand the associations between age and poor physical health days and, in doing so, we want to see if sex is either a confounder or effect modifier. We'll use the epiR library for this question.  
   1. Use the epi.2by2 function with the cohort.count method to estimate the crude RR to quantify the strength of the association between age 65 years older and 15+ days of poor physical health. Give the RR with 95% confidence interval (CI), and *p*-value. **(2 points)**

* 1. Let’s estimate adjusted and stratum-specific RRs:
     1. Use the epi.2by2 function with the cohort.count method to estimate the sex adjusted (and stratified) RRs to quantify the strength of the association between age 65 years older and 15+ days of poor physical health. **(2 points)**
     2. Make a table with the crude and adjusted RRs, including 95% Cis and *p*-values. **(2 points)**
     3. Would you consider sex to be a confounder, effect modifier, or neither? Explain. **(2 points)**

1. We’ve ignored the fact that our data have survey weights up to this point. Let’s address that now.  
   1. Use the svydesign function to define the survey design so R understands how to weigh the data (refer to the class slides on this). Be sure to configure options to allow for having a single observation per stratum (a “lonely psu”). We now have a clean data set and have created our survey design, so we’re ready to start our analysis. **(5 points)**
   2. Using svytable, create one-way frequency tables for the age group, age 65+, sex, 15+ poor physical health days per month. **(5 points)**
   3. Combine your svytable commands from part b with prop.table and cbind to create frequency tables that include counts and proportions for the same variables that you tabulated in part b. **(5 points)**
   4. Using svytable, create a 2x2 table of 15+ days of poor physical health x age 65+. Have age category be the rows, and health day category be the columns. What proportion of people had 15+ days of poor physical health per month in each of the two age categories? **(4 points)**
   5. Use svymean to estimate the mean number of days per month in poor physical health. What is the mean and its standard error? **(2 points)**
   6. Use svytotal to calculate the total number of days spent in poor physical health *in 2017*. What is the total number of days spent in poor physical health and the standard error of that total? **(2 points)**