PHPM 672/677 Lab #2: Variables & Conditionals
Due date: Submit by 11:59pm Monday 2/5 with Assignment 2

Overview

Most assignments will have a companion lab to help you learn the task and should cover similar content. You will be given time in class to work on the lab and ask me questions as you work through them. You should seek help from all sources (me, peers, internet) in doing this lab. **You do not need to document sources of help as this is a learning experience, rather than a graded homework.** However, it is your responsibility to complete the lab outside of class, if you were not able to complete them during the allotted time in class. There will not always be sufficient time in class to complete, but it is important that you complete them in order to learn the required contents.

Labs are part of the companion assignment. Labs will not be graded, but if you do not submit them 2 points will be taken off the companion assignment grade.

Submission. Labs are due on the same day as the companion assignments. Attach the following to the companion assignment submissions

- Commented code (SAS: lnameN_lab.sas; where N indicates the assignment number and lname is your last name)
- Output from your code (SAS: lnameN_lab.log & lnameN_lab.lst or lnameN_lab.html;)

Objective

In this assignment, you will learn the basics of working with one dataset. By the end of this assignment, you should be able to

- To write conditional logic codes
- Subset columns (variables) & rows (observations) from a given table
- Recode, rename variables and calculate new variables
- Label variables and values

Setting Up

1. Create a working directory, often referred to as pwd (present working directory), where you will work on this assignment. For example, "lab2". You will be writing code in lab2/
2. Create a sub directory data/. When you are writing programs, it is often a good idea to put your code in a different directory from your datasets, because if you are writing a lot of code, or working with a lot of datasets, you will stay more organized.

Getting Data: Download 2013 National Survey on Drug Use and Health (NSDUH) dataset which is conducted by SAMHSA and housed at the Inter-university Consortium for Political and Social Research (ICPSR) at the University of Michigan.

This dataset contains survey measures for the prevalence and correlates of drug use in the United States, such as information on illicit drugs, alcohol, and tobacco use, substance abuse treatment history, DSM diagnostic criteria, personal and family income sources and amounts, health care access and coverage, illegal activities and arrest record, neighborhood safety/context, and respondent demographics. **The goals of this homework are for you to download and import this data into SAS; clean and recode this data using the survey documentation as a guide; and some basic, descriptive patterns in the data as instructed below**.

Data housed at ICPSR are generally well maintained and documented, and they usually have SAS versions of the datasets available for download (which saves you from having to convert and label your data).

At this point, the website will ask you to create a free log in (probably only if you are on campus (if you are on campuses it should recognize your IP address on campus) and agree to terms of service.

Input

1. Download the required **SAS SETUP** files for SAS. DO NOT download the SAS links.
2. You should read the data use agreement, so you understand what is allowed with the data.
3. You should download something like ICPSR 35509.zip. You will need to unzip these into the lab2/. There is a subfolder (called \DS0001") that contains the dataset documentation and code required to setup the data.
4. copy the data file (DS0001/35509-0001-Data.txt) into data/nsduh.txt (renamed)
5. copy the required code file into lab2/
   a. DS0001/35509-0001-Data.sas into lab2/nsduh.sas (renamed)
   b. Edit the sas program as follows and take out comments for formats & missing

```
--- FROM (in the sas program given)
DATA;
INFILE "data-filename" LRECL=6836;
--- CHANGE TO (make this edits)
libname data "DIRNAME";
DATA data.nsduh;
INFILE "DIRNAME/nsduh.txt" LRECL=6836;
```

   c. Scan the program. See how they are doing things.
   d. Run your program (You do NOT need to submit the log/lst for this)
6. TEST: check the data directory and confirm that you have new sas dataset named as you did in your program (*.sas7bdat). Note this is also the output from this first step that will be the input to the next step. This would be an intermediate result. You need to finish up to this step before you can move onto the next step.

Use the Dataset

This is a large dataset - so the dataset should be used with care. If you load this dataset fully you will have:

```
obs: 55,160
vars: 3,141
size: 370MB
```

WARNING: This is a large dataset, DO NOT try to open the dataset as it will slow down your computer. If you want to see your data, use `proc print data=fn (obs=n);`

If you are lost and do not know how to use a particular syntax, look it up in the reference manual. Read the explanation, and in particular look at example code in the bottom. Go to the link below, click on index (give it time to load) and look for the keyword. Remember there are most always more than one way to do something, just find one that makes sense to you. At this phase, your only goal is to have the computer do things correctly.

Now work on the Problems P1 to P4 below with the dataset. Scan this section, so you understand what you are trying to do. Then, Read the **supplemental action plan** section (under P4) BEFORE starting to work on it, so you understand how to do this iteratively.

P1. Subset vars

SAS: use `keep` statements.

The following variables are what we will use for this assignment. They are indicators for substance use, treatment, insurance coverage, race, gender, health status, age, work status, and county of residence. `caseid, cigever, alcever, cocever, mjever, txever, medicare, caidchip, champus, prvhltin, income, health, irsex, catage, newrace2, coutyp2, wrkhrsw2`

P2. Describing the Data

P2.1. Describe the dataset, including the number of obs and vars that are in your subset of data.

P2.2 Check your data.

(a) Examine each variable for its range (min to max) or number of categories. SAS (`proc means, proc freq`)

(b) Discuss any anomalies you see in this data (such as an abnormally high or low values)

Be sure to read the codebook documentation provided by ICPSR. (Tell me why there are odd values/categories in some variables and not others)

P3. Cleaning and Manipulating the Data

Preparing your data for analysis usually involves:

- Adding or changing variable labels, variable names, and value labels;
- Checking and coding missing values;
- Calculating new variables or manipulating existing variables so that you can use them for analysis
  - SAS: assignment statements
- Checking and correcting the data formats (Next Lab);
- Reshaping and combining tables (Next Lab);

P3.1 Calculating new variables or manipulating existing variables

In this section you will calculate some variables that we can use in the next section for descriptive analysis

- Generate a new variable, called `subst`, that is equal to 1 if the respondent has ever had any substance in our sub-dataset (ever responded YES to using cigarette, alcohol, cocaine, marijuana) and equal to zero otherwise.
- Print a few observations to check that you have coded correctly. Make sure you check for different situations when checking.
- Generate a new variable, called `type_insur`, that is 0 if the respondent has no insurance of any kind, 1 if the respondent has government-based/public insurance (medicaid, medicare, or champus/tricare/va insurance), and 2 if the respondent has private insurance. If the person has both public and private insurance, code them as 3.
- Print a few observations to check that you have coded correctly. Make sure you check for different situations when checking. For example, coded as 3.

P3.2 Checking and coding missing values: `MJEVER`

- Recode `MJEVER` variables to be 1 for YES, and 0 for NO. It is currently coded 2 for NO. (SAS: assignment statements)
- Check the mean of `MJVER`. This should give you the percent YES. Does it?
- Use the codebook or setup programs to determine if `MJVER` have missing value codes. If so, make sure those values are coded as missing value codes so that they will affect analysis (like

calculation of a mean). Missing numerical values in SAS is '.', and strings are '' (NO space between quotes).

- Recalculate the mean of `MJEVER` you recoded for missing and compare with previous
- Now code missing properly for all variables, except the ever variables (which you will do below). I find using the setup programs very useful for this. But you can use any of the documents you downloaded.

P3.3 Calculating new variables or manipulating existing variables

Now you are going to repeat what you did above using DIFFERENT code. Learning different ways to do the same task is a good way to learn to code well. Some sections of the code can be the same as before.

- Create a new binary variable for all your "ever" variables by adding 'b' (for binary) as prefix of the same variable name (i.e., "`cigever`" to " `bcigever` ")
- Code all binary variables to be 1 for YES, 0 for NO, and missing for all other values. For analysis, this is the best way to code all binary variables.
- Labels your variables (SAS: `label`)
- Label your values (SAS: `proc format`)
- Drop the original "ever" variables (SAS: `drop`)
- Print a few observations and basic descriptives to check everything is correct.

P3.4 Adding or changing variable labels, variable names, and value labels

- Rename the `CASEID` variables to `ncaseid`, for numerical. (SAS: `rename`)
- Label the variable and values `subst` and `type_insur`
- Print a few observations and basic descriptives to check everything is correct.
- Display the variable labels for all variables in this dataset.

P3.5 Save out your data (i.e. into data/mynsduh) into a permanent dataset for use later.

- SAS: Do the above in a data step

P4. Learning Your Data (Descriptive Analysis)

In this section you will perform some descriptive analysis of continuous and categorical data in the subset of data.

P4.0 Identify variables by type : `proc contents;`

Look at the descriptive to initially determine variables by type

- Continuous variables
- Categorical variables (both ordinal and nominal)
- Binary variables (you should have explored this in the previous section)
- ID variables (not meaningful)

P4.1 Create a summary table for any continuous variables

Include the mean, standard deviation, p99, and N (total Number) values in this table for each continuous variable.

- SAS: `proc means`

P4.2 Create these cross-tabulations for categorical variables

Several commands can be used to examine categorical data. These commands can be used to examine descriptive/bivariate relationships between variables in the form of cross-tabulations.

- SAS: `proc freq; tables` *`var1 var2`* `/nocol norow nopercent`

P4.2a Ever Used any Substance - `subst`

Explore the relationship between `subst` and any 2 of the respondent characteristics (age, gender, income, county type, race/ethnicity). For instance, what percent and/or number of males have ever used any of the substances versus females? What are the differences between substance use across race/ethnicity categories?

P4.2b. Subset rows: Alcohol or Drug Treatment - `txever`

Answer the following: what percent of respondents who have ever used a substance and make $75,000 or more dollars have been to Alcohol/Drug Treatment?

- First subset the table to only those who have had substance use
- SAS: `where conditional;`

P4.2c. Overall Health Status - `health`

What is breakdown of Health Status (e.g. % Excellent, %V.Good, %Good, %Fair, %Poor) for respondents with the following characteristics/attributes: female, over the age of 25, they have used a substance, they are Hispanic or NonHispanic White, and they worked more than 25 hours a week?

Supplemental action plan

Step 1

- Now you will write your first SAS program.
- It will have one data step, and a few proc steps.

Step 1.1

- Open base sas
- In the editor window, write the first few lines of your sas code as below. **Note this is a different sas program than the one you edited above to generate the dataset. It might be named lname_lab2.sas.** Replace the unbolded words as appropriate for your program, if needed. You can use the same dataset name for indataset and outdataset, meaning that you will overwrite your old dataset with the new one you create. For this assignment, I suggest using a different name so you do not overwrite it. Note data.indataset should be the input to this step, and output from the previous step.

```
libname data "DIRNAME";
data data.mynsduh;
set data.nsduh;

proc print data=data.mynsduh(obs=10);
proc contents data=data.mynsduh;
run;
```

- Save your program into lab2/
- Run this program in sas. Check the log. It should tell you that now you have a new dataset named with the new name, with the same number of observations and variables. You should have no ERRORs or WARNINGS.
- TEST1: look at the output (results). Check that things look correct.
- TEST2: check in the data directory for the new dataset. See that it is the same size as the old dataset. Note this will change in the next step.

Step 1.2 Subset vars

- Now go back to your editor with your three four lines of code.
- Add lines of code in the data step (between `data … ;` and `run;`) to do one thing. For example, to subset variables (keyword `keep`).
- Now rerun ALL the sas code. Look at the log. This time you should have a message telling you that you only have 17 variables
- TEST: look at the output. Does this look right?

Step 1.3 Describing the data

- Now go back to your editor with your code.
- This time, you will add proc steps after your data step to describe your data (`proc contents, proc means, proc freq`). Put these between your `proc print` and run statements
- Again check your log
- TEST: check your output

Step 1.4

- Iterate over these steps adding in one task at a time either to the data step (step 1.2), or adding in a new proc step (step 1.3).
- After each task is completed, remember to save your program, always rerun the code, look at the log, and look at the results.
- If you need to stop for the day, save your program and come back to the particular task next time.

Output: Submission

You DO NOT submit the datasets. I don't need to see your datasets, because I can see your programs. I know exactly what you did to your datasets.

The best way to submit, is after you are all DONE.

- Save your program.
- Exit out of SAS (this is easier than clearing the log and results window, because by now you will have a LOT there).
- Start up SAS again.
- Open your sas program
- Now run your full code. (this should give you a clean log and results)
- Save out the clean log and results (*.html)
- Submit your program, log, and results.