# DATA MANAGEMENT

# DATA MANAGEMENT

- Missing values
- Recoding
- Lab5
- Secondary variables
- Grouping variables
- lab 6

# DATA MANAGEMENT

- Data management involves making decisions about recoding data in ways that help answer the research questions.
- The code book and frequency distributions are an excellent guide for making these decisions.
- You will use frequency distributions often as they reveal things about the data that will be helpful.
- It is possible that your variables will already be adequately managed and no additional data management will be necessary.

# DATA MANAGEMENT

- It is important to consider each step in the data management process as it allows you to play an active role in understanding your data and assuring that you are asking your questions in an appropriate and meaningful way.
- Step 1 is to consider if you need to code out missing data.
- Using the NESARC data and focusing on young adult smoking we can see that the response category 1 means daily smoking for variable S3AQ3B1
- The remaining response categories are also in the codebook for the usual smoking frequency variable.

# DATA MANAGEMENT

- Your data will often include response categories that do not help answer your question even though they provide information.
- For instance, in the variable S3AQ3B1 there is a response category of unknown, coded as 9.
- For those responses we simply don't know how much these individuals smoked, and therefore we don't want to include them in our analysis.
- Python ignores missing values if they are set to null or nan

# MISSING VALUES

- In Python, specifically Pandas, NumPy and Scikit-Learn, we mark missing values as NaN.
- Values with NaN are ignored from operations like sum, count etc
- We can change the values to a NaN value using the Pandas DataFrame replace() function on a subset of columns we are interested in.
- After we change the values, we can use the isnull() function to confirm through counts that the missing values were marked as NaN

# MISSING VALUES

- Useful tips in python:
  - dataset.describe() outputs summary statistics on each variable (mean, std, min, quartiles and max)
  - dataset.head(20) outputs the first 20 rows of data. Any number can be passed to the function.
  - (Dataset[1,2,3,4,5]==0).sum()) counts the number of zero values in the variables named 1,2,3,4,5

# MISSING VALUES

- dataset[[1,2,3,4,5]]=dataset[[1,2,3,4,5]].replace(0, numpy.NaN)

- In the dataset variables 1,2,3,4 and 5 have their values of zero replaced with NaN.
- A count on the number of nulls should show the same as a previous count of zero values.

- print(dataset.isnull().sum())

# MISSING VALUES

- Having missing values in a dataset can cause errors with some machine learning algorithms.
- Pandas provides the dropna() function that can be used to drop either columns or rows with missing data.

- dataset.dropna(inplace=True)
- This will remove all rows that contain an NaN.

# RECODING VALUES

- Imputing missing values refers to using a model to replace missing values.
- There are many options:
  - A constant value that has meaning within the domain, such as 0
  - A value from another randomly selected record
  - A mean, median, or mode value from the column
  - A value estimated by another predictive model
- Pandas provides the fillna() function for replacing missing values with a specific value.

# RECODING VALUES

- A survey often has questions that the respondent can skip over. This can create missing values. Sometimes we can recode the missing values as valid data.
- In the NESARC survey the question "did you drink at least one alcoholic drink in the past 12 months" was answered as No by 16,116 respondents (S2AQ3)
- These individuals would then not need to answer the question about how often they drank. (S2AQ8A) Instead it was set to blank or NaN.
- It would be reasonable to code this as valid data rather than missing, we do this by placing an 11 where there is a blank in S2AQ8A and a no answer in S2AQ3.

# RECODING VALUES

- data.loc[(data['S2AQ3']!=9) & (data['S2AQ8A'].isnull()),'S2AQ8A']=11

- the loc[] function accesses the data using the conditional statements all in one go, if you don't use loc pandas can get confused and your operation fail.

# RECODING VALUES

- Another useful step in data management is to give your variables response codes that may be more logical than those they were originally given.
- For the usual smoking frequency variable we can see in the code book that lower values mean that the respondents smoked more and higher values mean that the respondents smoked less
- We can choose to reverse code this variable so that higher values mean more smoking, and lower values mean less smoking.

# RECODING VALUES

```
recode1 = {1: 6, 2: 5, 3: 4, 4: 3, 5: 2, 6: 1}
subset['USFREQ'] = subset['S3AQ2B1'].map(recode1)
```

- format for a recode is {old value : new value}
- We also gave the variable a new name, USFREQ so it is easier to remember and so we don't confuse it with the original variable S3AQ3B1