Data Cleaning Process

Loading Process

First, loading the csv file to the data frame. Thankfully, the data was presented in such a format where it was relatively easy to read into a pandas dataframe. Immediately from looking at it, I could see that there was both missing data and zeroes in the data. One other huge problem was when I loaded the data into the dataframe, that was just raw values, and no column names associated with it. There was an additional list at the bottom which contained the column names, but not in a format easily read into a list. Therefore to merge with the existing dataframe ended up being a multistep process.

Merging

The format of column names needed to be manipulated first before being able to merged. I grabbed the list, placed it in excel, then separated out the data using Excels "Text to Columns" function, which allowed me to pull just the column names, and separate out other extraneous characters including the definitions of said columns, which I know that I'll need when I do the final data analysis and wrap up, but is not necessary for data analysis right now, and therefore was excluded. With a little bit of massaging, I was able to create a CSV of column names, which I then uploaded to the Python notebook and saved as the original dataframe's column names.

Dealing with Missing Values

I noticed when I was first importing the data that there were missing values in the data which seemed to be denoted by "?"s. After in fact checking that they were strings of question marks, I then wrote a function to count the number of ?s(missing data in this case) to count the number of columns with missing data. I noted after running this function that there were a few data sets that jumped out at me, with large amounts of missing data. One of these was a per capita column, where my intention is to fill that in with a possibly weighted average of other per capitas in that community area using the average of other races in the area to weight them. One other staggering thing I noticed was that of the 2215 rows of information I had, 1872 of them were missing police data, a whopping 84.5% of total police data was denoted as "?". With that much missing data, I did not feel comfortable wanting to keep that in my data set, and instead eliminated the columns entirely. Trying to make predictions based on such a small pool would most likely lead to inaccurate results. I decided to drop the 22 columns that had the vast majority of the data missing, most of which had to do with police census type data, and decided to just predict with the other 100+ columns. I used a manual drop, though I could have used a function to find where it said police or polic, since the spelling used was inconsistent, I decided to do it manually as to not accidentally keep or drop wrong columns.

Dealing with Irrelevant Data

Although this was already denoted when I first retrieved the data set, it wasn't actually clear what the irrelevant data was going to be until I successfully put the column names back into their associated columns. At that point, it was then I saw that there were going to be identifying variables, which would help in the end stages, but most likely hinder early data analysis. I knew that they would not need to be

included in the analysis, but I did not believe they would need to be removed entirely. These columns included the community name, county code, and community code.

In addition, it was noted in the data gathering portion that the last 18 columns/variables would be the ones to be predicted, but I did see that they already had data in them. This could be useful to check how accurate the algorithm is at the end, but serve no purpose in the early data analysis, nor in the machine learning portion. While I plan on leaving them in the dataframe for the time being, they will not be used in the final product, other than to compare accuracy.

Checking Data Values

I began doing this by first grabbing the columns that would be the easiest to check. For me, this was all the columns containing a percentage of some sort, which all began with “pct” which made it very easy pull all relevant columns into a new dataframe. At this point I wanted to make sure that the values were all between 0-100. Anything outside of those boundaries would be incorrect data and would need to be omitted. I wrote a np.where to check where the percentage would’ve been outside of these boundaries and an empty array was returned to me, so all data within that percentage dataframe was within acceptable boundaries. Will be doing similar things with the other columns.

In addition, I used a bit of graphical analysis to try to find outliers, and it was difficult to figure out down all columns so I will be trying to narrow down, and using boxplots and histograms to find outliers that would severely skew the data and eliminating them from the dataset.