|  |  |  |
| --- | --- | --- |
|  |  | Poverty  vs  clearance rates  By Zoe Halbeisen  SI 330 Final Project |

4/17/17

# introduction

My project involves analyzing FBI homicide reports and Census Bureau poverty estimates for states in the US. I will be looking at clearance rates (percentage of murders that are solved) and poverty rates by county to see if there is a correlation between low clearance rates and poverty levels.

I am proposing this project because I was inspired by an article I read about the Murder Accountability Project which is a nonprofit project started by Thomas K. Hargrove. The project is “dedicated to educate Americans on the importance of accurately accounting for unsolved homicides within the United States.” Hargrove also created an algorithm that can help law enforcement catch a serial killer. I think Hargrove’s work is extremely interesting and an awesome example of how data can be used to make a real difference and save people’s lives. I am also shocked at how many unsolved homicides there are in the United States. This assignment is a great way for me to get involved in Hargrove’s project and to use the datasets he has compiled.

I want to take a different take on it by comparing clearance rates with poverty rates. I choose these factors to compare because I am curious how class status affects homicide data. This question is spurred by the Flint Water Crisis in which I and many others believe race and poverty had a lot to do with the decisions that led to the water crisis. I wondered whether that injustice for poor communities could also be extended to low clearance rates, aka more unsolved murders, in poor communities. I want my program to take a stab at this issue and test whether there is in fact a correlation.

# data sources

I used two data sources for my project: FBI Uniform Crime Reports & U.S. Census Bureau’s Small Area Income and Poverty Estimates.

## FBI Uniform Crime Reports

**Description:** Using the Murder Accountability Project website, the data compiled includes two datasets maintained by the Federal Bureau of Investigation: The Uniform Crime Report (I only used 2015) and the Supplementary Homicide Report. These datasets include extensive details about homicides such as the victims race, gender, and weapon of choice. Since I was interested in clearance rates, I extracted the agency, clearance rate, county, murder count, state, and year fields.

**Size:** 3251 records/293kb

**Location:** <http://www.murderdata.org/p/blog-page.html>

**Format:** CSV file  
**Access Method:** The Murder Accountability Project has the data compiled on their website and I filtered results for 2015 only and then downloaded the data to a csv file.

**U.S. Census Bureau's Small Area Income and Poverty Estimates (SAIPE)**

**Description:** The U.S. Census Bureau's Small Area Income and Poverty Estimates (SAIPE) program provides annual estimates of income and poverty statistics for all school districts, counties, and states. The main objective of the program is to provide estimates of income and poverty for the administration of federal programs and the allocation of federal funds to local jurisdictions. In addition to these federal programs, state and local programs use the income and poverty estimates for distributing funds and managing programs.

**Size:** varies based on API call

**Location:** API Base URL - <http://api.census.gov/data/timeseries/poverty/saipe>

**Format:** JSON   
**Access Method:** I fetched data from the Census Bureau using their API and urllib.request

# Data processing steps

**Step 1: Fetch data from the Census Bureau using their API and urllib.request**

This happens in the census\_bureau\_SAIPE function. When called this function makes the request and gives back data depending on which state code gets passed in. The data comes back as JSON format. The function’s output is a nested dictionary with keys of county names and values being dictionaries with keys poverty rate and poverty count.

**Step 2: Read in the Uniform Crime Report csv file using csv.DictReader**

This happens in the read\_ucr function. This function’s input is a file and a string representing the desired state. The output is a nested dictionary with keys of county names and values being dictionary with keys murder and clearance count. This function also acts as a counter and has to distinguish whether a county has already been encountered as the original csv file is ordered by Agency, so counties are duplicated.

**Step 3: Clean the data and find matching counties**

This happens in the clean\_CB and clean\_UCR functions. Both of these functions’ inputs are dictionaries, which are the same dictionaries outputted from the census\_bureau\_SAIPE and read\_ucr functions. These cleaning functions match their syntax to find counties that appear in both datasets. This is necessary because reporting to the FBI UCR is voluntary so there are some counties in the Census Bureau data that is not in the FBI UCR data. The counties that are in both and had 5 or more homicides in 2015 are then stored into the list “match.” The reason that I had to filter out counties with less than 5 homicides was because there were many counties with very low homicide numbers which skewed the data to show higher clearance rates.

**Step 4: Write out data to new csv file using csv.DictWriter**

Using the master dictionaries I created, I use these and the matches list to write out data to a new csv file, results.csv. I combine the Census Bureau data with the FBI data to output County, homicide count, solved homicide count, clearance rate, and poverty rate. In this block of code I compute the clearance rate by dividing the solved homicide count with homicide count. I wrote a helper function, convert to percentage, to make it into a percentage. I use a try/accept clause when writing these out because in some cases the Census Bureau data does not follow its standard naming convention. For example, it’s usually “Washtenaw, County” but in some rare cases like with Baltimore City, it’s just “Baltimore City” with the “, County” part omitted.

**Step 5: Optimize program to take user input**

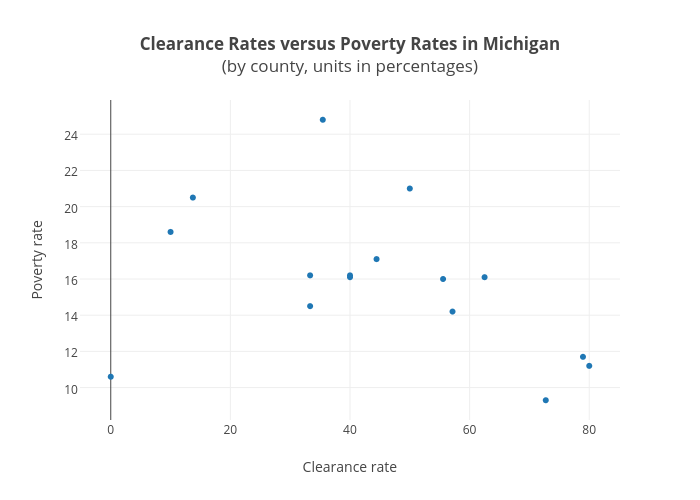
I wanted my program to be able to be used as a tool and to be able to run quickly on any state. Before, I had to manually change the API call to put the correct state code and manually type in the desired state and abbreviation following the conventions of the data sources. So, to make my program take user input, I first had to read in a text file of state codes which I got from the Census Bureau site. I did this because a normal user would not know the state code but would rather just type “Michigan.” The state codes are then stored into a dictionary, state\_codes\_dict, which I map the user input values to. I also create a state\_abrev\_dict dictionary which stores the abbreviations for each state. The variables state\_code, state, and UCR\_state are also created to be used throughout my program.

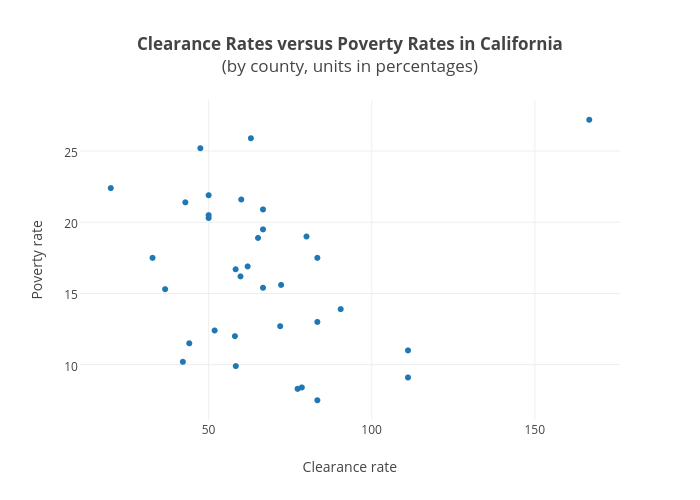
**Step 6: Create scatterplot with Plotly**

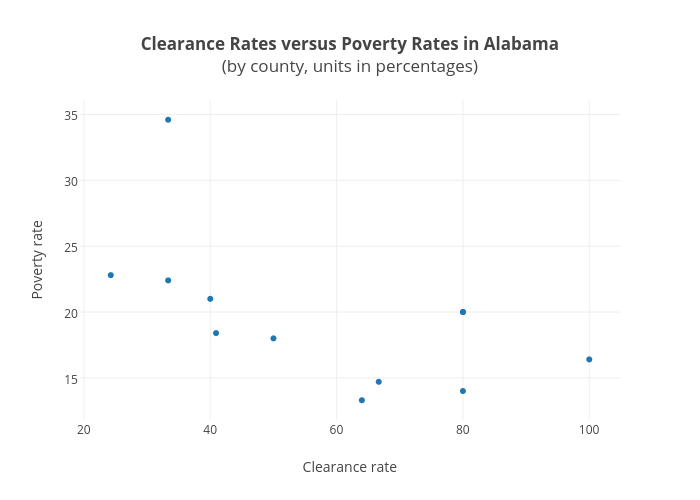
Lastly, I wanted my program to create a scatterplot of clearance rates vs poverty rates automatically. I used Plotly to accomplish this. I use my master dictionaries and then the output is a scatterplot which opens in the browser.

# results

My initial question was to see whether there is a correlation between clearance rates and poverty rates. I found that there is indeed a general negative trend for most states where there was enough data. Here are some example scatterplots which were all created by my program and Plotly.







We can see that there is indeed a negative trend, meaning that counties with higher poverty rates tend to have lower clearance rates, or more unsolved homicides.

# discussion & Conclusion

I think that the negative trend between poverty and clearance rates is alarming, and more research should be done to draw any substantial conclusions. If I had more time to work on this project, I would work on gathering more data. My program collected data from 2015 only, so I think expanding it to encompass 5-10 years would be very insightful. I also think considering other factors besides poverty rates, for example, population size or race, would be necessary as well since it is difficult to draw conclusions from just two variables. I think the issue of injustice for the poor is way more complex than just looking at clearance rates. Having more advanced visuals would also help in finding patterns. I would love to have the data output to a map somehow since the data is based on geographic locations. I am also not sure if everything I did followed the correct standards and protocols of data analysis. For example, I just filtered out counties with less than 5 homicides, but I am sure there is a more correct way to deal with this. I would also learn how to deal with outliers. I am not as informed on statistical analysis or know exactly if I followed all the rules.

Overall, this project taught me how useful and diverse data analysis can be. I also realized that data analysis can be fun when you are working on datasets that you are interested and passionate about. I found myself wanting to finish my project and make it better because I was genuinely curious and interested in the question I was addressing. This was also a great final project because it tied together concepts from over the course and forced me to think how I could apply them. The most challenging part was the beginning trying to figure out exactly what methods and formats would be the best for what my end goal was. Our homework assignments always came with a template and told us what tools to use, so this was a great final project that represented how data analysis would be conducted in the real world.

# references

Link to Bloomberg article on Murder Accountability Project:

<https://www.bloomberg.com/news/features/2017-02-08/serial-killers-should-fear-this-algorithm>

Python CSV File Reading and Writing documentation:

<https://docs.python.org/2/library/csv.html>

Plotly Python Library documentation:

<https://plot.ly/python/>

Census Data API User Guide:

<https://www.census.gov/data/developers/guidance/api-user-guide.html>