Is West Virginia Poor?

An Economic Analysis of West Virginia and Her Counties

**Abstract**

This report examines data on poverty, income, and unemployment, focusing specifically on West Virginia and how poor it is compared to the rest of the US. It also compares geographically the counties that are poorest based on several factors. The data was analyzed using maps of the counties in West Virginia and maps of the US allowing comparisons between counties in West Virginia and between states. This study found that West Virginia, when compared to the other states in the US, has the seventh highest percent of citizens below the poverty level and the third lowest average income. It furthermore finds the counties in West Virginia that are the poorest based on poverty, income, and unemployment and finds that they are geographically close. This study shows that West Virginia is poor compared to most of the states in the US.

**Introduction**

Despite the economic power that the US has, there are many poor areas in the US. West Virginia is commonly considered one of the poorer states in the US. I analyzed data on poverty, average income, and unemployment to come to a conclusion on whether West Virginia is truly a poor state. I plotted the levels of those above variables in different counties and compared West Virginia graphically to the rest of the US. Furthermore, I analyzed the counties in West Virginia that are the poorest to learn if there is a geographic correlation between those counties.

**Data**

I used three datasets for this project. The first was a large dataset for every county in the US with economic data, population data, and electoral data from both the 2016 and 2020 presidential elections. There are a total of 50 columns in this dataset. However, I only used seven of those columns in my project, focusing on specific population and economic data that are commonly associated with being poor. There are records of every county in the US and Washington DC. This dataset is located here on GitHub: <https://github.com/Deleetdk/USA.county.data>. It was accessed on April 23, 2023, but I cannot locate it there anymore. Nevertheless, the link takes you to a GitHub page that has other similar datasets with the same data but more variables that are unnecessary for this project.

The second dataset that I used was a dataset with FIPS codes and county names. The FIPS codes were necessary for plotting the data of counties using the choropleth plot from plotly.express as they correlate to the location of the county. There are FIPS codes for every county in the US and Washington DC. This dataset was accessed on April 23, 2023 and is located here on Kaggle: <https://www.kaggle.com/datasets/danofer/zipcodes-county-fips-crosswalk>.

The final dataset that I used was a JSON file that contains the county lines for all the counties in the US. All I did with this dataset was give it to the choropleth plots to form the maps. This dataset was accessed on April 23 2023 and is located here: <https://raw.githubusercontent.com/plotly/datasets/master/geojson-counties-fips.json>.

**Methods**

First, I imported pandas, plotly.express, json, and urlopen from urllib.request.

Beginning with the CSV file that contains county names and FIPS codes, I used read\_csv() to create a pandas data frame out of it. This data frame I called “county\_fips.” I only used the columns for county name, FIPS code (called “STCOUNTYFP”), and state as none of the rest are necessary for this project. Then I called pipe() on the data frame two separate times.

1. The first function given to pipe(), one\_of\_each\_county(), removes all the duplicate county entries because the data frame originally had entries for every town/city in the US. It removed the duplicate counties by returning the given data frame after it was grouped by FIPS code and a function that I called “first\_row” was applied to it. The function first\_row() takes a data frame and uses iloc to just return the first (index 0) row of the dataframe. In summary, one\_of\_each\_county() simplifies the data frame so that it only has one row for each county in the US.
2. The second function given, no\_county\_at\_end(), creates a list out of the column of county names. Then, it goes through every element in the list using list comprehension and if the name ends in “County” that element in the list gets set to its name without “County” at the end. The same is done with “Parish” as all of Louisiana’s counties end in parish” instead of “county.” In summary, this function removes phrases that are commonly after county name so that the data frame can be merged with our other set of data which does not have these phrases after the names of the counties.

For the CSV file that contains the bulk of the county data I also used read\_csv(), using only the columns for county, state, population, employed people, poverty, income, and unemployment as none of the rest are necessary for this project. No other changes are necessary for this data frame. This data frame I called “county\_statistics.”

After the two files are read correctly and their data frames are cleaned, the next step is to use merge() to create a combined data frame. I called merge() on county\_statistics, giving it county\_fips. I merged left and right for the data frames’ respective names for county and state and used an inner join. I had to merge on both county name and state because there are some counties with the same name but in different states. After the merge, I renamed the column “STCOUNTYFP” to “fips” as that is much more easily readable. I then called pipe() on this merged data frame twice.

1. The first function I gave to pipe() on this merged data frame was state\_poverty(). This function takes a data frame. It creates a series from grouping by state on the original dataframe and using apply() on poverty and population from that grouped by data frame. The function given to apply, weighted\_poverty(), first sums up the total population. Then every individual poverty level is multiplied by that county’s individual population divided by the total population of all the counties (in the state, because of the group by). Then all those individual values are summed together to get a weighted poverty level for each state. A new column is then created for the data frame that we will return, called “StateAvgPoverty,” which I set equal to a list comprehension getting each state’s data from the series created earlier and rounding it to two decimal places based off the state of the county. That data frame is finally returned, giving another column with the average poverty for the entire state.
2. The second function is state\_income. This function also takes a data frame. It creates a series from grouping by state on the original dataframe and using apply() on average income and total employed from that grouped by data frame. The function given to apply, weighted\_income(), first sums up the total employed. Then every individual average income is multiplied by that county’s individual amount of employed people divided by the total population of all the counties (in the state, because of the group by). Then all those individual values are summed together to get a weighted average income for each state. A new column is then created for the data frame that we will return, called “StateAvgIncome,” which I set equal to a list comprehension getting each state’s data from the series created earlier and rounding it to two decimal places based off the state of the county. That data frame is finally returned, giving another column with the average income for the entire state.

I called this combined data frame “combined\_county\_data.”

I also created another data frame, combined\_wv\_county\_data, and used boolean indexing to select only the counties with state values equivalent to “WV.”

To get the county geography in a form that can be given to a choropleth plot, I used urlopen on the link to the JSON file mentioned in the data section of this report and called it response using the form “with urlopen(‘X) as response:” (with X being the actual url given in the Data section of this report). Then I set a variable name “counties” equal to “json.load(response)” which takes in a JSON file so that it can be used later on.

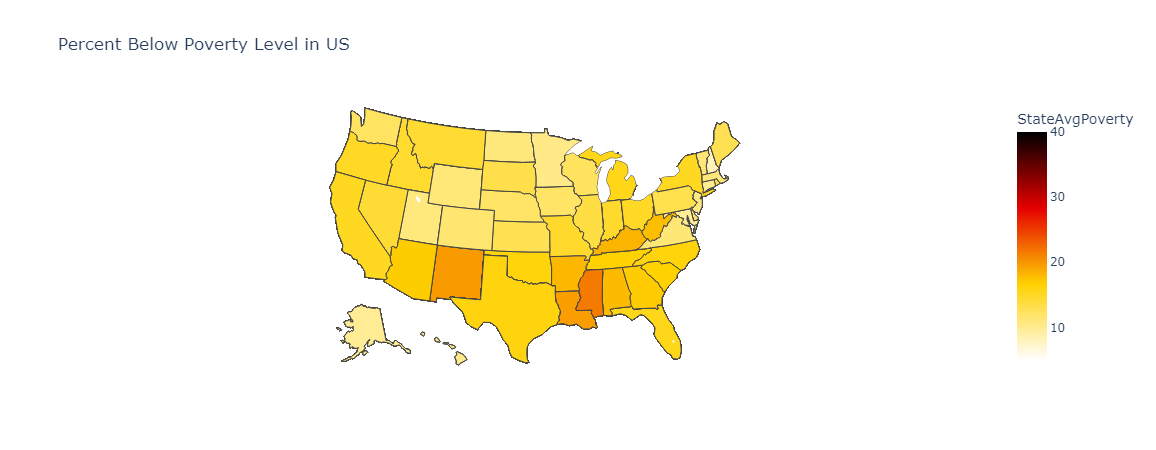
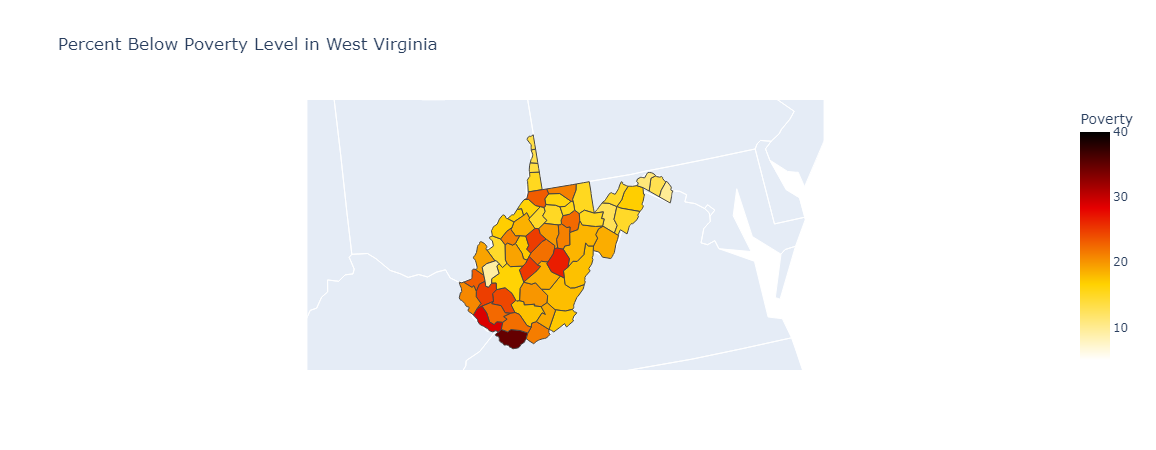
**Results**

*Question 1 – What percent of people in West Virginia are below the poverty level and how does that compare to the rest of the US?*

For generating the plots for question one, I used a choropleth plot from plotly.express. For producing the West Virginia plot, I gave it the combined\_wv\_county\_data, but for the US plot I gave it combined\_county\_data. For the different arguments, locations I set equal to the FIPS code column (for West Virginia) or state column (for US), hover\_data to county, color to poverty, color\_continuous\_scale to “Hot\_r” (the “\_r” reverses the colors, which makes more sense for this plot), range\_color to the range from 5 to y (with y being 40 for the West Virginia plot and 25 for the US plot as the percent of people below the poverty level can be more extreme in counties than entire states), scope to “usa,” and title to “Percent Below Poverty Level in X” (with “X” being either “West Virginia” or “US”). Furthermore, I set locationmode equal to “USA-states” for only the US plot and geojson equal to the county variable created from the JSON file for only the West Virginia plot. Also, before showing the West Virginia plot, I called updategeos(fitbounds = ‘locations’) on it to zoom into West Virginia so that it does not show the entire US.

I created a sorted list of the states based off poverty by grouping combined\_county\_data by state and finding the max of “StateAvgPoverty” (but because they are all the same for the same state, it just returns one instance of that state). Then I sorted the values, setting ascending equal to False, reset the index and just selected the state column. This I then converted to a list. So, to find West Virginia’s rank, I called index() on the list, giving it “WV” and adding one to that. I found West Virginia’s average poverty level just by taking the index 0 column of combined\_wv\_county\_data and selecting “StateAvgPoverty” from that.

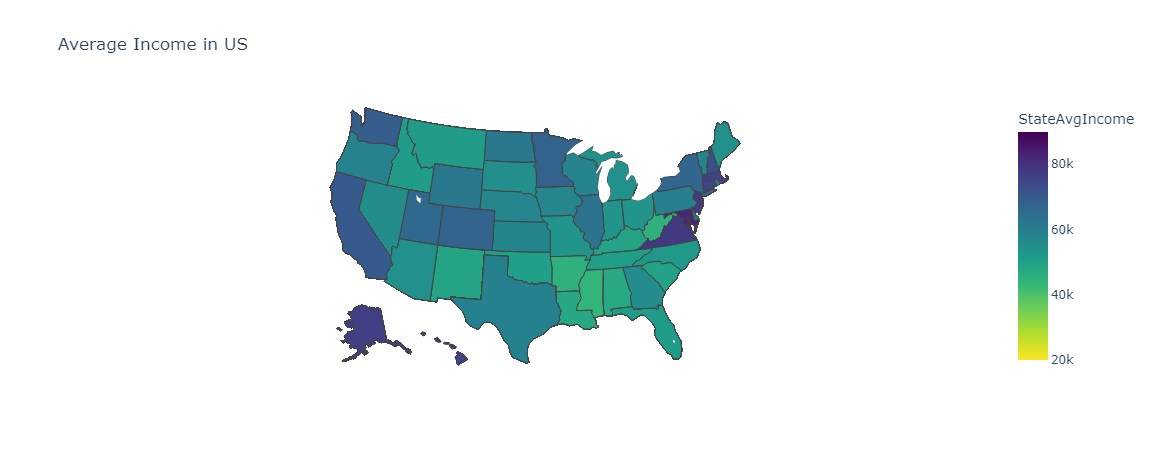
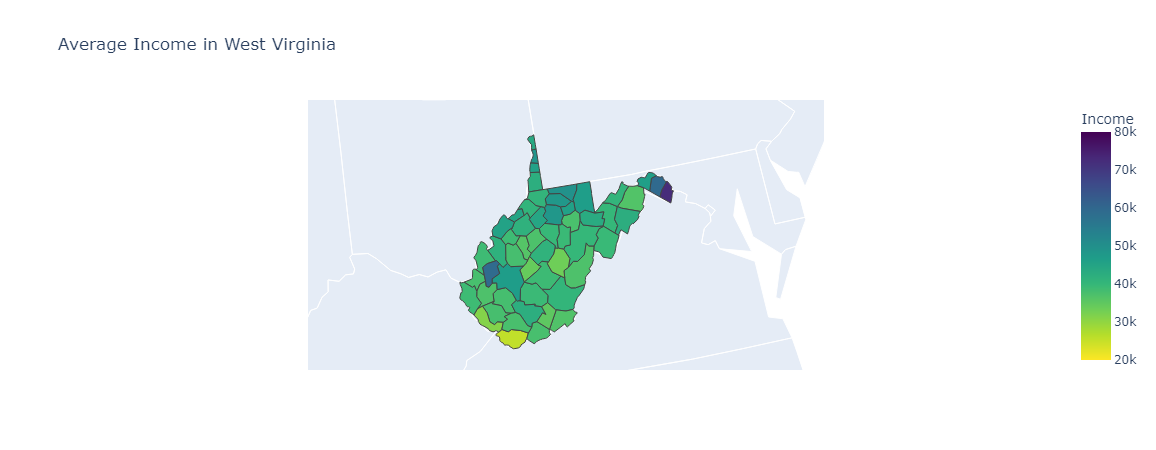
I found that West Virginia has the 7th highest percent of citizens below the poverty level among US states and Washington DC with 17.8%.



*Question 2 – What are the average incomes in individual counties in West Virginia and how does West Virginia's average income compare to the rest of the US?*

Similarly for question two, the plots were created the same way for this second question as they were for the first with the following differences. The argument color was set equal to income instead of poverty, color\_continuous\_scale to “Viridis\_r,” and the title said, “Average Income” instead of “Percent Below Poverty.” Also, range\_color was set to the range from 20000 to 80000 for the West Virginia plot and from 40000 to 90000 for the US plot.

Finding how West Virginia’s average income ranked compared to other states was done exactly like it was for question one but using “StateAvgIncome” instead of “StateAvgPoverty.”

I found that West Virginia has the 3rd lowest average income among US states and Washington DC with $45620.13.

*Question 3 – Is there a correlation between the 20 West Virginia counties that have the largest percent of people below the poverty level, the 20 counties with the smallest average incomes, and the 20 counties with the largest unemployment rates? If so, which counties are they, are they geographically close, and are nearby counties in other states also poor?*

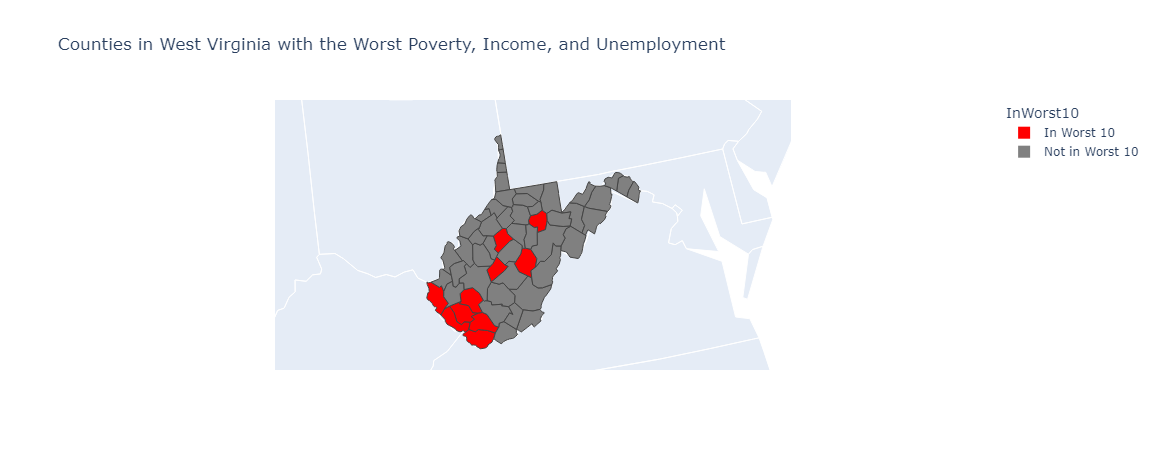
For question three, I began by creating three ordered lists, one each for poverty, income, and unemployment. I first selected only the columns for county and X (with X being either poverty, income, or unemployment) from combined\_wv\_county\_data. Then I sorted values on X (setting ascending equal to False for poverty and unemployment, but True for income as bad poverty and unemployment are large numbers whereas bad income is a low number). Then I used head() to select the first 20 of these rows, then selected just the county column. Finally, I converted these to lists.

Then I created an empty list, wv\_p\_i\_u\_list, and looped through every county in the poverty list created above. If that county was also in the income and unemployment lists, then it was appended to wv\_p\_i\_u\_list. To display the list without the brackets inside a formatted string that I printed, I used join() on “, ” and gave it all but the last element of list using slicing. Then after an “and” I inserted the last element of the list. Also, I made a variable called worst\_counties that is the length of the wv\_p\_i\_u\_list.

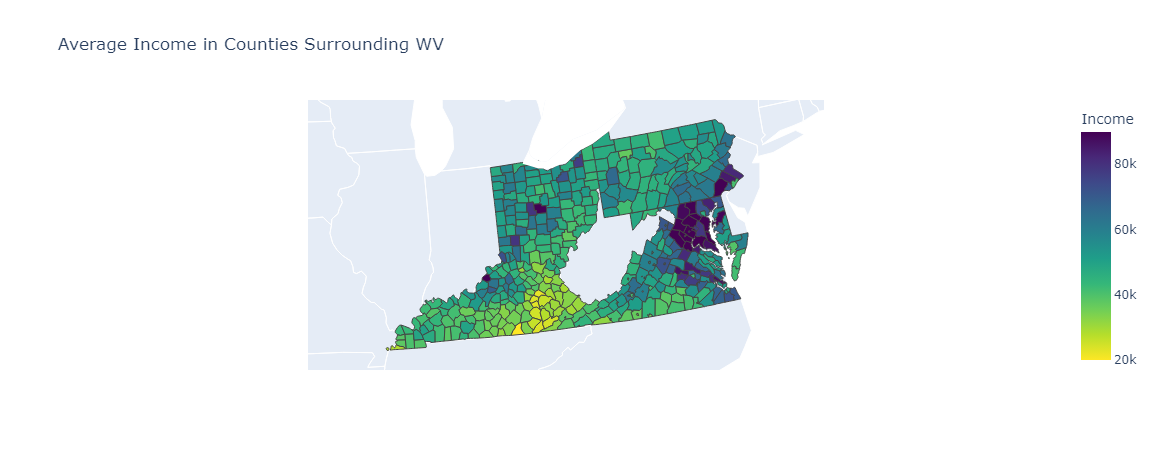
I then created a data frame, wv\_p\_i\_u, from just the FIPS code and county column from combined\_wv\_county\_data. I created a new column for whether a county is in the wv\_p\_i\_u\_list, calling it “InWorstZ” (with Z being the variable worst\_counties). I created this column using list comprehension, going through each count in wv\_p\_i\_u and if it is in wv\_p\_i\_u\_list, setting its value in the new column equal “In Worst Z” and otherwise setting its value to “Not in Worst Z.” To allow a variable name in a string, I used f before the strings to allow for formatting. For this question, I was unable to use pipe() as I created lists that I used to create the new columns and those lists are also needed for the rest of this step.

I created a choropleth plot again, setting all the arguments to the same values except for the following: I set color equal to “InWorstZ” (continuing with Z’s definition form the last paragraph), did not have color\_continuous\_scale but had color\_discrete\_map equal to a dictionary assigning “In Worst Z” to “red” and “Not in Worst Z” to “grey,” and set title equal to “Counties in West Virginia with the Worst Poverty, Income, and Unemployment.” As before I called updategeos(fitbounds = ‘locations’) on it to zoom into West Virginia.

For the surrounding counties in other states I created a data frame that I called “combined\_wv\_surrounding\_area\_county\_data” and created by using boolean indexing on combined\_county\_data such that the state was either “MD,” “DC,” “VA,” “PA,” “OH,” or “KY” (the states surrounding West Virginia and Washington DC, where there would have been a hole in the plot). Then I created a choropleth plot, giving it combined\_wv\_surrounding\_area\_county\_data and setting geojson equal to counties, locations to FIPS codes, hover\_data to a list with both county and state, color to income, color\_continuous\_scale to “Viridis\_r,” range color to the range from 20000 to 90000, scope to “usa,” and title to “Average Income in Counties Surrounding WV.” Before showing it, I called update\_geos(fitbounds='locations') on it.

I found that there are 10 counties in WV that are among the worst 20 counties when it comes to all three of poverty, income, and unemployment. Those counties are McDowell, Mingo, Webster, Clay, Gilmer, Boone, Barbour, Logan, Wyoming, and Wayne County.

After analyzing the plot, there does appear to be some geographic correlation between the counties with the largest percent of people below the poverty level, the lowest average income, and the highest unemployment rates.



There do seem to be many poor counties (based off average income) in other states surrounding the poor counties in south-western West Virginia, especially counties in Kentucky.

**Conclusion**

I analyzed this data in order to find out if West Virginia is truly poor. When it comes to poverty, West Virginia has the seventh highest percent of people below the poverty level when compared to the rest of the states in the US and Washington DC, which is clearly not good. Furthermore, West Virginia has the third lowest average income, which would seem to indicate that West Virginia is a very poor state. It can be seen from the first three West Virginia plots that central and southwestern West Virginia are the poorest areas. This is confirmed by the third West Virginia plot (the West Virginia plot in Question 3), which shows the counties that are among the worst 20 when it comes to all of poverty, average income, and unemployment.

More research should be done, then, to find out why those counties are so poor and if there is anything feasible that could be done to help those counties. Also, research could be done to see how these variables have changed over time to see if the situation these counties are in is getting better, worse, or staying the same.

**Appendix**

To validate my data, I plotted it using choropleth plots. I graphed all of population, poverty, income, people employed, and unemployment for just West Virginia counties. I also checked the variables that I used on the US plots using the same method. This is how I found the issue with Louisiana’s counties ending in “Parish” instead of “County” as there was no data showing up for Louisiana and the two data frames were not merging their data on Louisiana.

From those plots I can see if any of the data is invalid by changing what I give to color\_range to below and above what I would expect and seeing if any counties or states have numbers beyond what is probable. After doing this for both West Virginia and the US as a whole, the data appears very consistent with no irregularities that I could detect.

The plots of variables mentioned above that were not plotted on maps in the Results section of this report are given below, showing that none of the data appears to be unreasonable (the county with the large population contains Charleston, the capital of West Virginia, which is the largest city in West Virginia). A picture containing diagram

Description automatically generatedA picture containing map

Description automatically generatedA picture containing diagram

Description automatically generated