Final Report

2024 US Census Data

Mitchell Sydlowski

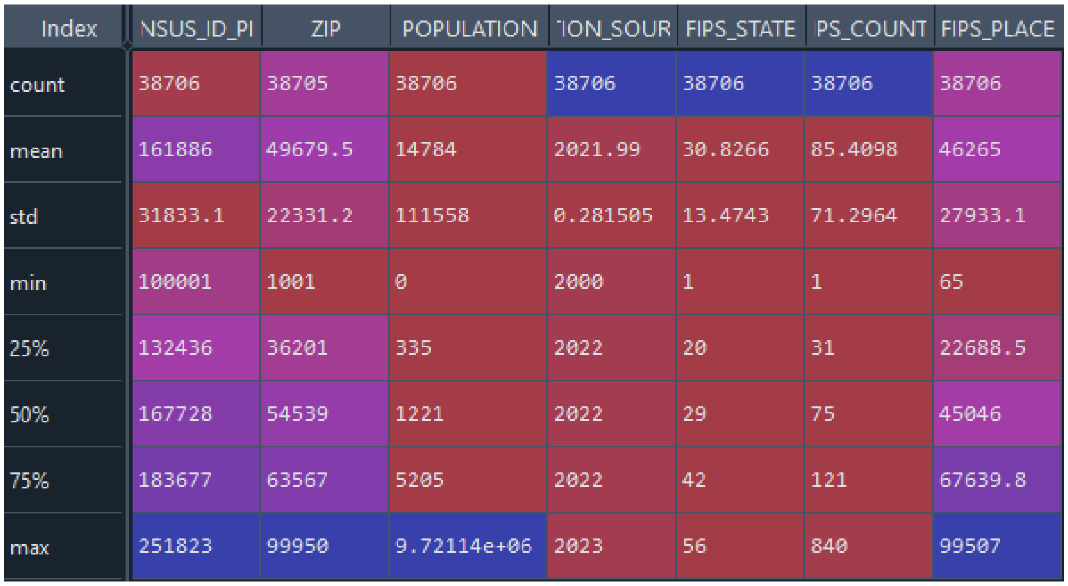
University of Mount Union

DSC 140

December 6, 2024

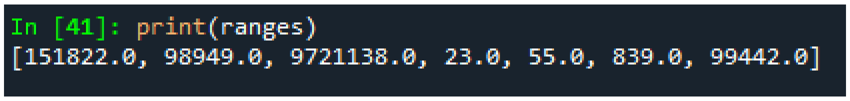
I chose to do my final report on the 2024 US Census data, formally from 2022 but saved under 2024 by the government, because it comes with a large amount of data entries and a sufficient number of columns to work with. The project consists of required analysis and then two research questions were constructed to further the data analysis. All analysis was conducted within Spyder utilizing Python code.

The required analysis consists of finding the mean, range, and standard deviation of numerical columns, Finding and describing relationships between columns of continuous and categorical data, this is data that contains text or is not classified as numerical, and applying a machine learning algorithm. First the data was imported using the pandas function, read_excel and saved under "data." The mean and standard deviation were found by using the describe function, data.describe(). This reported a mean and standard deviation for each column of numerical data. It also reported a maximum and minimum value, the difference was then calculated to report the range for each column. This can all be found in **Figure 1 and Figure 2** below.



Index	NSUS_ID_PI	ZIP	POPULATION	'ION_SOUR	FIPS_STATE	PS_COUNT	FIPS_PLACE
count	38706	38705	38706	38706	38706	38706	38706
mean	161886	49679.5	14784	2021.99	30.8266	85.4098	46265
std	31833.1	22331.2	111558	0.281505	13.4743	71.2964	27933.1
min	100001	1001	0	2000	1	1	65
25%	132436	36201	335	2022	20	31	22688.5
50%	167728	54539	1221	2022	29	75	45046
75%	183677	63567	5205	2022	42	121	67639.8
max	251823	99950	9.72114e+06	2023	56	840	99507

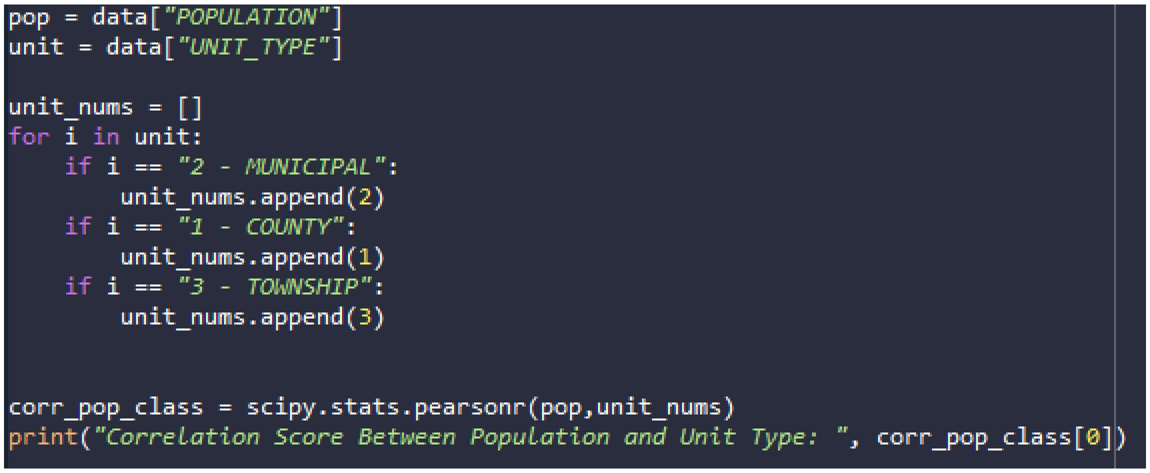
Figure 1: Mean, Standard Deviation, Max and Min Values



```
In [41]: print(ranges)
[151822.0, 98949.0, 9721138.0, 23.0, 55.0, 839.0, 99442.0]
```

Figure 2: Ranges for Numerical Data, Columns Same as Figure 1

The next task was to determine and describe relationships between columns of continuous data. The two relationships I looked at were Unit Type and Population and Political Code Description and Population. For both analyses, both Unit Type and Political Code Description needed to be classified numerically. This was done with the code shown in **Figure 3**. This shows the process for changing the data entries for Unit Type from “1 - County”, “2- Municipal”, and “3 - Township”, into 1, 2, and 3. The process was repeated for Political Code Description, for sake of space within this report I have decided to omit showing the code for this reclassification as there as 21 different options within the column. Please refer to the included code if you wish to see the process for Political Code Description.



```
pop = data["POPULATION"]
unit = data["UNIT_TYPE"]

unit_nums = []
for i in unit:
    if i == "2 - MUNICIPAL":
        unit_nums.append(2)
    if i == "1 - COUNTY":
        unit_nums.append(1)
    if i == "3 - TOWNSHIP":
        unit_nums.append(3)


corr_pop_class = scipy.stats.pearsonr(pop,unit_nums)
print("Correlation Score Between Population and Unit Type: ", corr_pop_class[0])
```

Figure 3: Code for Numerical Reclassification of Unit_Type

The last two lines in **Figure 3** show the process of retrieving and reporting a correlation score between the two columns of data. A correlation score ranges from –1 to 1. The closer the score is to –1 or 1, the more correlated or “dependent” the data is. The closer to 0, the less correlated and “independent” the data is. The correlation score for Unit_Type and Population was –0.163. Meaning it has a very low, negative correlation. I determined that the score was low enough to state that there was no correlation between unit type and population. The correlation score for Population and Political Code Description was –0.109, I also determined that there was no correlation between these two columns as well. These two sets of columns were then graphed to visualize the population values for each unit type and political code description. These graphs can be found in **Figure 4 and Figure 5**.

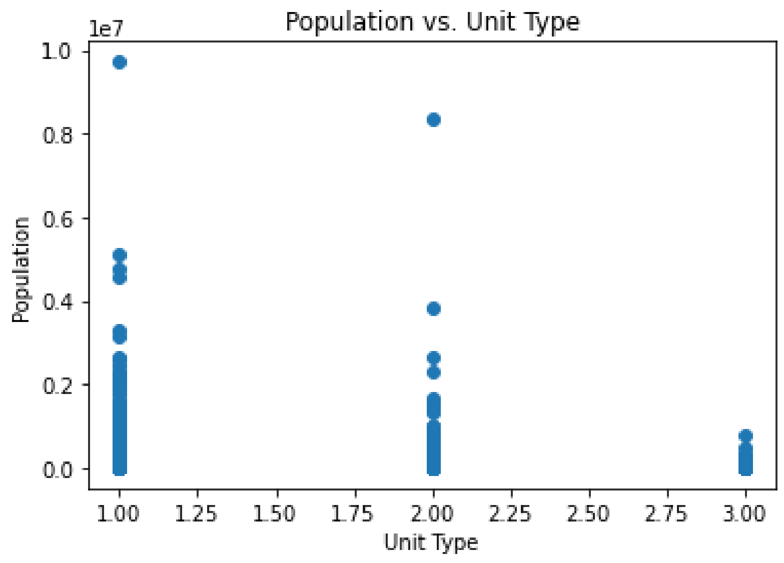


Figure 4: Population vs. Unit Type

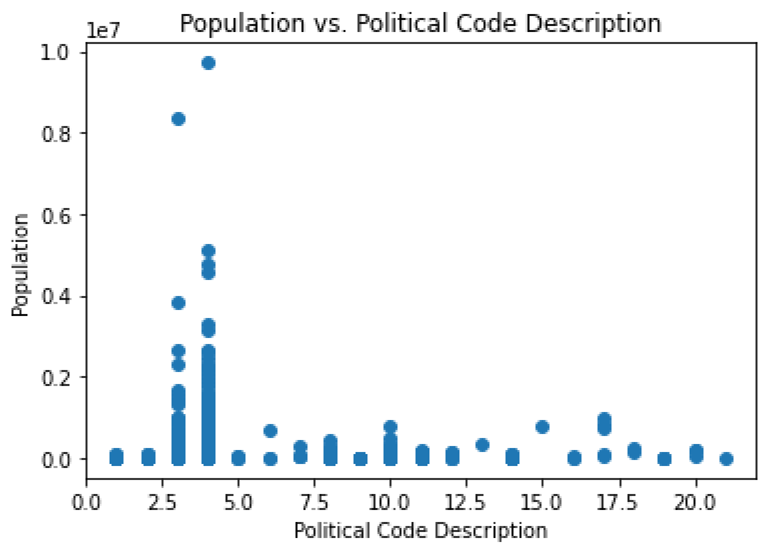
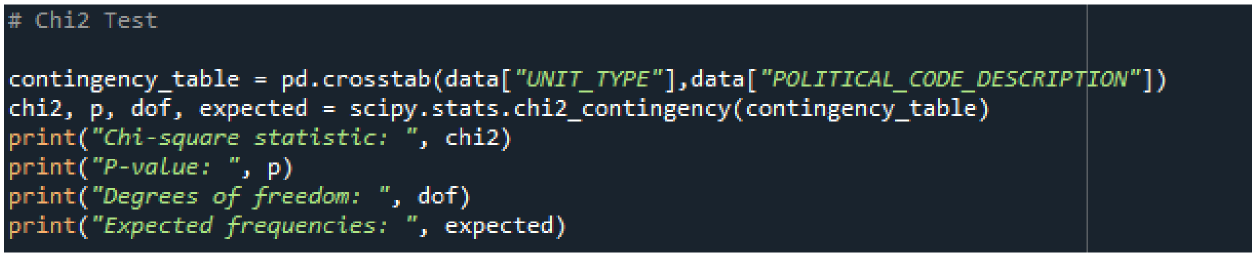


Figure 5: Population vs. Political Code Description

Although there seems to be little to no correlation between population and either Unit Type or Political Code Description, the graphs show that population typically remains around the same for every classification and the larger populations can be found when observing counties as a whole, or cities. This is to be expected as zooming out to the county level will raise the population count and observing urban major cities will also yield large population numbers due to the density in these areas.

The next portion of required analysis was to describe a relationship between two columns of categorical data. This was performed through a chi-square test. **Figure 6** displays the code to perform this test. This test reports a high chi2 statistic of 68,611. This means we observe a significant difference between the observed and expected data.



```
# Chi2 Test

contingency_table = pd.crosstab(data["UNIT_TYPE"],data["POLITICAL_CODE_DESCRIPTION"])
chi2, p, dof, expected = scipy.stats.chi2_contingency(contingency_table)
print("Chi-square statistic: ", chi2)
print("P-value: ", p)
print("Degrees of freedom: ", dof)
print("Expected frequencies: ", expected)
```

Figure 6: Code for Chi-Square Test

Next a machine learning algorithm was constructed to predict the classification for Unit Type based on Population and Political Code Description. This was done with k-nearest neighbors, this is a machine learning process that takes the neighboring data entries to try and predict the output for a desired entry. The number of neighbors utilized for this process was 1, this was chosen by running the algorithm with multiple neighbor values to determine which was the best. A confusion matrix was then constructed to display the number of correct and incorrect predictions made by the model. This is shown in **Figure 7**. Each time the process is run the accuracy score fluctuates a bit. Typically, the accuracy fell somewhere between 77 – 78 %.

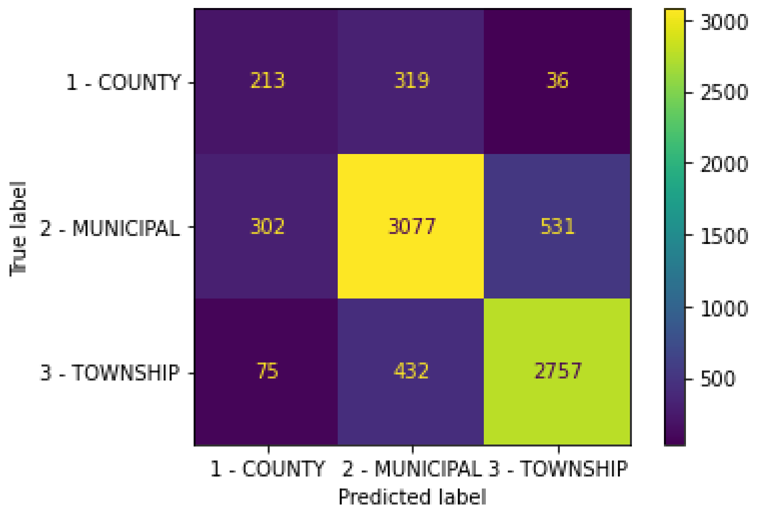


Figure 7: Confusion Matrix for Machine Learning Algorithm

The next portion of the project were my two research questions. These were: “which counties hold the highest and lowest percentage of the population for each state?” and “Based on population and potentially various other variables can a Machine Learning process correctly predict which type of classification each data point is?” This first question was answered by masking the data down to only include entries classified by county within the Unit Type column, splitting the entries up by state, and then calculating percentages to determine which county in each state holds the largest and smallest percentage of the state's population. There is no included figure for this section within the report, refer to lines 173 - 218 and run the code. The print statement runs through a for loop and does the calculations for each state without saving the data. The second question was completed by repeating the process for the prior machine learning algorithm. This time however, Political Code Description was classified by Population, Unit Type, and Zip Code. The confusion matrix can be seen below in figure 8 and the code for this portion can be found from lines 221 – 251. The accuracy score for this algorithm was 69.63% with a k-nearest neighbors value of 13. Again, this value was determined by running the code with a for loop that changes the neighbors value and prints the accuracy score, the highest score and corresponding neighbors value was then utilized.

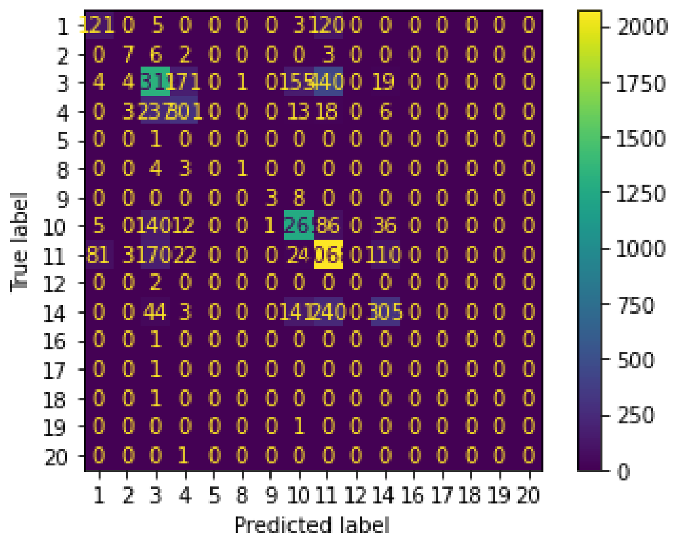


Figure 8: Confusion Matrix for Classifying Political Code Description