CS 564 Final Visualization Project

Hailee Kiesecker

Fall 2020

01 Introduction :

For my final visualization project for CS 564 I decided to create a visualization for computer science general users to use in making an informed decision when going to work in the field after college. The main. For a lot of recent college graduates they are going into their field of interest and do not fully understand how much their education is worth. In the Information technology field it seems to be common to see huge pay gaps for different jobs offered at different companies. You can google it and find a basic answer: on average $71,500 (Forbes). However there are still technology start up companies hiring recent graduates at $40,000 a year (Interview). So what are our computer science, recent graduates really worth and how much can they actually make outside of academia?

Our solution to students and recent graduates not knowing how much they should negotiate their pay based on their job is through data visualization of an interactive United States map.

02 Data Set :

I accessed my data set through the US bureau of Labor Statistics (BLS). This is a public dataset collected by the department of labor, established by President Taft in 1913. The BLS is a statistical agency that cleans, analyzes, and spreads its collected statistical information to the general public, and government agencies.

For the purposes of this project I wanted a dataset that would be able to show users what would be considered normal for a job within the computer science field. The "`May 2019 National Industry-Specific Occupational Employment and Wage Estimates`" dataset provided an industry specific data field along with a variety of wage estimates for each job type. Using python I was able to extract key information that I needed for my visualization of showing how each US state compares to each other in minimum, average, and maximum salary ranges. In Table x I list all of the attributes within our dataset; name: '`State_M2019_dl`' .

Field	**Field Description**
area	U.S. (99), state FIPS code, Metropolitan Statistical Area (MSA) or New England City and Town Area (NECTA)

	code, or OES-specific nonmetropolitan area code
area_title	Area name
area_type	Area type: 1= U.S.; 2= State; 3= U.S. Territory; 4= Metropolitan Statistical Area (MSA) or New England City and Town Area (NECTA); 6= Nonmetropolitan Area
naics	North American Industry Classification System (NAICS) code for the given industry
naics_title	North American Industry Classification System (NAICS) title for the given industry
i_group	Industry level. Indicates cross-industry or NAICS sector, 3-digit, 4-digit, 5-digit, or 6-digit industry. For industries that OES no longer publishes at the 4-digit NAICS level, the “4-digit” designation indicates the most detailed industry breakdown available: either a standard NAICS 3-digit industry or an OES-specific combination of 4-digit industries. Industries that OES has aggregated to the 3-digit NAICS level (for example, NAICS 327000) will appear twice, once with the “3-digit” and once with the “4-digit” designation.
own_code	Ownership type: 1= Federal Government; 2= State Government; 3= Local Government; 123= Federal, State, and Local Government; 235=Private, State, and Local Government; 35 = Private and Local Government; 5= Private; 57=Private, Local Government Gambling Establishments (Sector 71), and Local Government Casino Hotels (Sector 72); 58= Private plus State and Local Government Hospitals; 59= Private and Postal Service; 1235= Federal, State, and Local Government and Private Sector
occ_code	The 6-digit Standard Occupational Classification (SOC) code or OES-specific code for the occupation
occ_title	SOC title or OES-specific title for

	the occupation
o_group	SOC occupation level. For most occupations, this field indicates the standard SOC major, minor, broad, and detailed levels, in addition to all-occupations totals. For occupations that OES no longer publishes at the SOC detailed level, the “detailed” designation indicates the most detailed data available: either a standard SOC broad occupation or an OES-specific combination of detailed occupations. Occupations that OES has aggregated to the SOC broad occupation level will appear in the file twice, once with the “broad” and once with the “detailed” designation.
tot_emp	Estimated total employment rounded to the nearest 10 (excludes self-employed).
emp_prse	Percent relative standard error (PRSE) for the employment estimate. PRSE is a measure of sampling error, expressed as a percentage of the corresponding estimate. Sampling error occurs when values for a population are estimated from a sample survey of the population, rather than calculated from data for all members of the population. Estimates with lower PRSEs are typically more precise in the presence of sampling error.
jobs_1000	The number of jobs (employment) in the given occupation per 1,000 jobs in the given area. Only available for the state and MSA estimates; otherwise, this column is blank.
loc quotient	The location quotient represents the ratio of an occupation’s share of employment in a given area to that occupation’s share of employment in the U.S. as a whole. For example, an occupation that makes up 10 percent of employment in a specific metropolitan area compared with 2 percent of U.S. employment would have a location quotient of 5 for the area in question. Only available for the state, metropolitan area, and nonmetropolitan area estimates; otherwise, this column is blank.

pct_total	Percent of industry employment in the given occupation. Percents may not sum to 100 because the totals may include data for occupations that could not be published separately. Only available for the national industry estimates; otherwise, this column is blank.
h_mean	Mean hourly wage
a_mean	Mean annual wage
mean_prse	Percent relative standard error (PRSE) for the mean wage estimate. PRSE is a measure of sampling error, expressed as a percentage of the corresponding estimate. Sampling error occurs when values for a population are estimated from a sample survey of the population, rather than calculated from data for all members of the population. Estimates with lower PRSEs are typically more precise in the presence of sampling error.
h_pct10	Hourly 10th percentile wage
h_pct25	Hourly 25th percentile wage
h_median	Hourly median wage (or the 50th percentile)
h_pct75	Hourly 75th percentile wage
h_pct90	Hourly 90th percentile wage
a_pct10	Annual 10th percentile wage
a_pct25	Annual 25th percentile wage
a_median	Annual median wage (or the 50th percentile)
a_pct75	Annual 75th percentile wage
a_pct90	Annual 90th percentile wage
annual	Contains "TRUE" if only annual wages are released. The OES program releases only annual wages for some occupations that typically work fewer than 2,080 hours per year, but are paid on an annual basis, such as teachers, pilots, and athletes.
hourly	Contains "TRUE" if only hourly wages

	are released. The OES program releases only hourly wages for some occupations that typically work fewer than 2,080 hours per year and are paid on an hourly basis, such as actors, dancers, and musicians and singers.

For our cleaned data all we needed to extract was all a_pct10, a_pct25, a_pc75, a_pct90, occ_code, and area_title. I then feature engineered a main minimum salary from using the occ_code column selecting the occ_codes for the computer and mathematical occupations list i.e. 15-0000 and column row values of a_pct10. Then averaging the minimum values per area_title i.e. state name. I did the same process to get the maximum salary using column row values of a_pct90. Finally to get the average salary I found the average per unique area_title using all annual salary percentile. Creating a CSV file that could cleanly be used with my d3.js visualizations for the interactive United States map.

02.1 Issues:

The main issue when dealing with this data was missing data, thankfully it did not occur too often. I had to make a giant work around in my python cleaning script to identify empty fields due to the changing of types not changing to work correctly. In the end the solution was to just make the missing data fields, as not to drop them because their other fields had useful information, to make the missing value either be the last known minimum or the last known maximum, depending on the feather being created. With that we end up with data as seen below:

```
id,state,min,max,avg

0,Alabama,41710,137950,84717.5

1,Alaska,50350,120560,81968.5
```

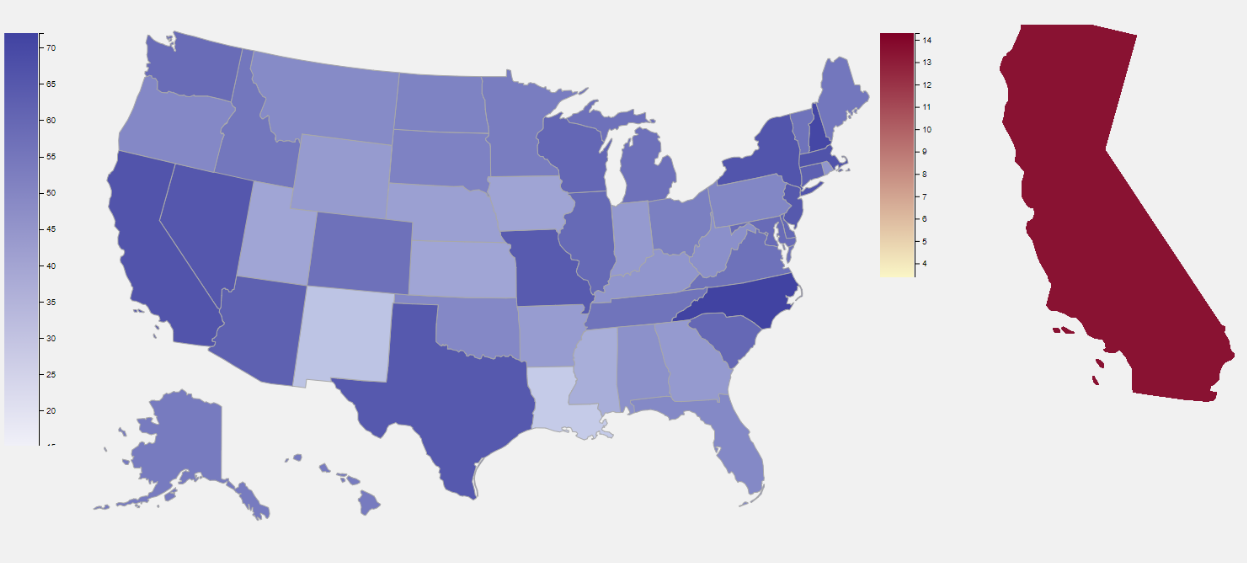
To help users make better informed decisions if they wanted to use this visualization as a basis to decide which state they would like to work in, I used "Unemployment Rates for States" recently updated in October 2020. I then added this information for each state to my clened_state.csv which now looks like the data below:

```
id,state,min,max,avg
```

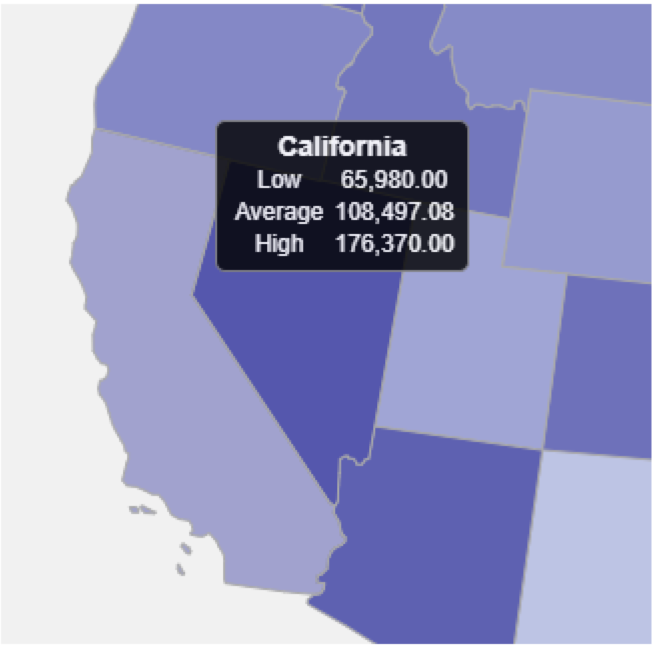
```
0,Alabama,41710,137950,84717.5,5.8

1,Alaska,50350,120560,81968.5,5.9
```

03 Architecture ：

For my visualization solution I wanted to take the cleanest approach possible, this visualization isn't meant to be the only source of information to computer science students looking for what to expect from their salary. But rather, a place to get a key idea on what states that would like to be looking at for employment and around how much they will be paid in that state. I created a basic US states map that allows the general user to hover over each state and see its general IT pay information. On top of that when a user clicks a state they are able to see the unemployment rate of that state, so if they ever want to start a startup or leave their new job, they have a guesstimate on how hard it will be to find work again after. View the figure below for a sample of the visualization;



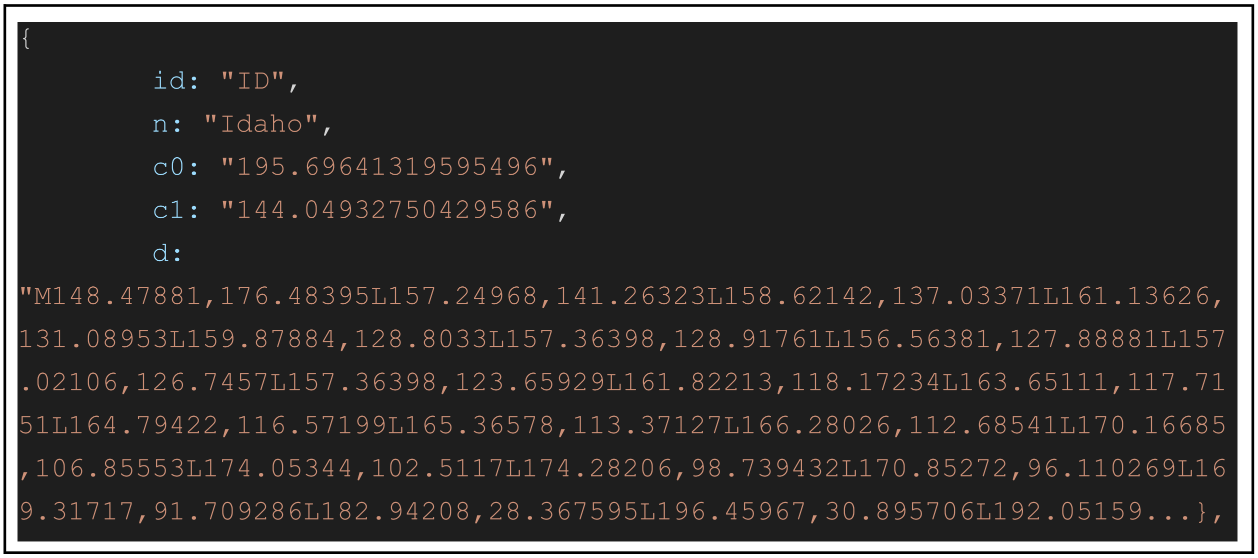
The legends are explained within the webpage and when the state is hovered over you can see its general salary information for full time IT related jobs. You can notice in the above image that the Blue legend (showing the minimum average salary for each state) is larger than the red legend (showing unemployment rates), I chose this instead of them being the same size because the main purpose of this visualization is the pay of an IT related professional. It takes up more of the page so that a user may easily find it while going through the visualization. I chose red and blue color legends because they appeared to have the biggest contrast factor with colorblindness able to identify the key differences between these two colors.



California

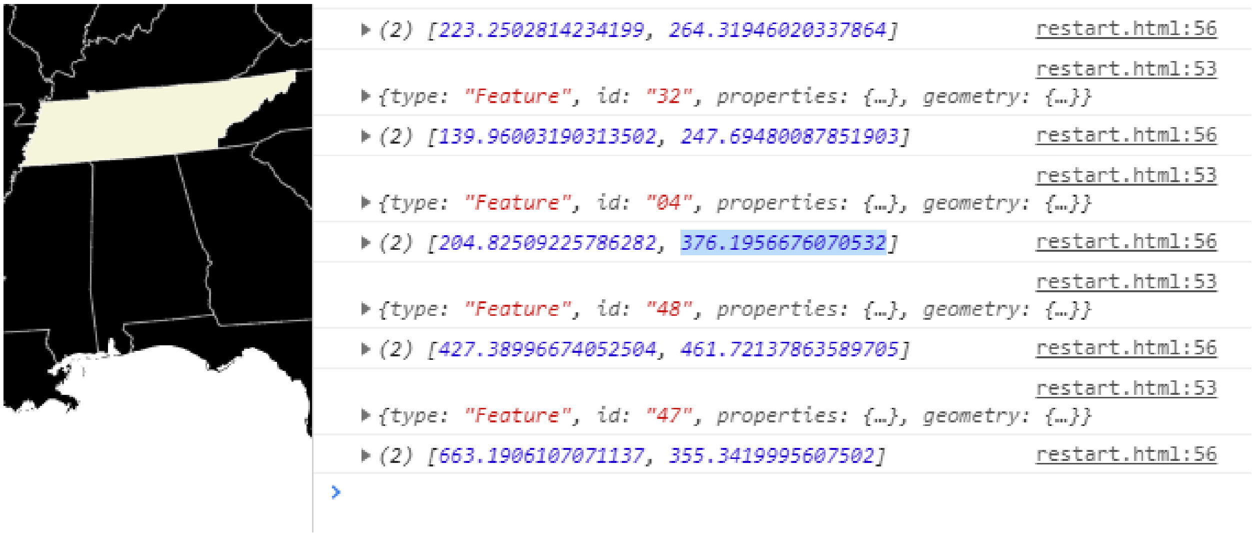
Low 65,980.00
Average 108,497.08
High 176,370.00

Issues in implementing the unemployment part of the visualization were getting the states when clicked to show up in the same middle area of the svg. Figuring out the centroids to use for each state was also difficult for how I had set up my original full state map. I ended up having to create a miniature webpage to extract each state centroid individually and add it to our javascript state information. View a sample bellow:



```
{
        id: "ID",
        n: "Idaho",
        c0: "195.69641319595496",
        c1: "144.04932750429586",
        d:
"M148.47881,176.48395L157.24968,141.26323L158.62142,137.03371L161.13626,
131.08953L159.87884,128.8033L157.36398,128.91761L156.56381,127.88881L157
.02106,126.7457L157.36398,123.65929L161.82213,118.17234L163.65111,117.71
51L164.79422,116.57199L165.36578,113.37127L166.28026,112.68541L170.16685
,106.85553L174.05344,102.5117L174.28206,98.739432L170.85272,96.110269L16
9.31717,91.709286L182.94208,28.367595L196.45967,30.895706L192.05159...},
```

By clicking each state individually from our mini webpage we manually got the c0 and c1 values.



```
▶ (2) [223.2502814234199, 264.31946020337864]                   restart.html:56
                                                                 restart.html:53
▶ {type: "Feature", id: "32", properties: {…}, geometry: {…}}
▶ (2) [139.960031903l3502, 247.69480087851903]                  restart.html:56
                                                                 restart.html:53
▶ {type: "Feature", id: "04", properties: {…}, geometry: {…}}
▶ (2) [204.82509225786282, 376.1956676070532]                   restart.html:56
                                                                 restart.html:53
▶ {type: "Feature", id: "48", properties: {…}, geometry: {…}}
▶ (2) [427.38996674052504, 461.72137863589705]                  restart.html:56
                                                                 restart.html:53
▶ {type: "Feature", id: "47", properties: {…}, geometry: {…}}
▶ (2) [663.1906107071137, 355.3419995607502]                    restart.html:56
>
```

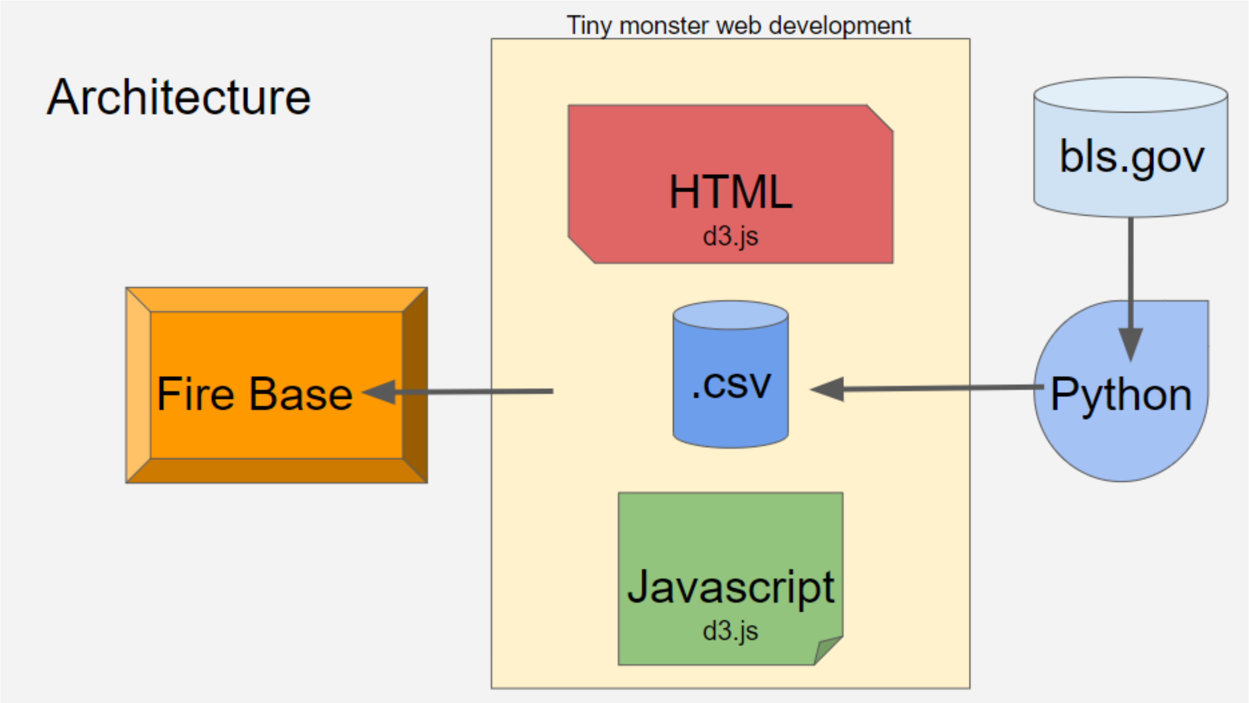
Another challenge brought about with this visualization was the legends and transposing them correctly within the frame of information. Initially they continued to be on top of each other no matter what. However, diving deeper with the inspection tool I was able to see that their svg's were individual, away from the main US state svg. Changing around key pieces of code, I was able to make their svg's be created within the main state one.

Lastly, the color of the states within the main US states map is being created by the minimum value of all US states salaries being compared against each other. It took a lot of trial and error to get this fill function to work. A Lot of the time whenever a few states were colored, others would be completely empty. It was a simple fix that took a lot of playing around with to get the numbers correct. It ended up being the minimum average salary divided by around 1,000,000. This is not common sense to me. I anticipated 10,000 or 100,000 one million seems illogical but once printing out the values it began to make more sense.

03.1 Design :

The initial design of the project was to have the visualization also include the influx of workers/employees within the business sector be shown in a line graph below each state. This however began to pose an annoying problem. Within my source of information, the Bureau of Labor Statistics makes it incredibly difficult to get multiple state's information easily. You must go into each state to download the data and the formatting of it is unusable to merge. If I have time to implement this feature I will, however, I truly do not think it adds much value to the users experience of this visualizations purpose.

In the following figure I outline how I chose to implement this visualization. As stated, I used the Bureau of Labor Statistics webpage (bls.gov) to get my data. I would then create python scripts that were able to process and clean the data that I had downloaded from bls's website. Finally my html, javascript, d3.js code would modify and manipulate the data to appear on my webpage visualization in the form that I thought brought the best user experience. Finally this webpage is being hosted by FireBase, a google hosting platform.



04 Conclusion :

In conclusion, I greatly enjoyed this project compared to the other ones in the class. I would work on it for fun and would get lost in it for hours. The only issue I'm having is the sheer amount of time it takes me to modify, understand and implement d3 code. I believe I am getting better at it but this project so far has taken me around 30 hours of work to complete, not including any documentation, and the amount of visualization that is displaying seems to be not equivalent to that work.

I am proud of this visualization. While it does not show multiple graphs of information such as a bar plot, or line plot, with the US state's, I believe it accomplishes its purpose of informing the user what they should expect pay wise for when they leave with their computer science degree depending on the state. If I were to continue to work on this visualization I would have displayed the highest paying job in that state along with the percentages of employment increases for the business sector of that state. As it stands I am very tired of trying to mess with the Bureaus data and am calling it an accomplished semester.