In the activity, “Google\_Geocode”, we were introduced to the Google Maps API. I learned how to use the .format method when building ‘target\_urls’ by using place holder numbers and then .format([parameter], [parameter]). We started out by using the ‘geocode’ url to gather basic information about a city in the US. We built on this idea by using the ‘nearbysearch’ url to discover places near the geographical coordinates of a place. In the “Google\_Places” activity, I learned how to add specific dictionary of params to be used in request.get of data from Google Maps API. The following student activity, “Google\_That”, I combined the combined the two concepts, ‘geocode’ and ‘nearbyserach’, to find bicycle stores near Seattle, Washington, balloon stores by the White House, and dentists near me in Chicago. I like how setting the params outside the request.get() makes the code look a lot neater by just using the url, params in the method.

The next part of class focused on how to iterate through a dataframe and then use the data to interact with the Google Maps API and try/except patterns. In the “Nearest\_Restr” activity, I learned how to use the ethnicity of a restaurant in one dataframe, grab information based on that type from the Google API, and then add the additional restaurant information to the same dataframe. We used the pd.loc() function again to place the information in the right column of the dataframe. In the student exercise, I learned how to apply both ‘geocode’ and ‘nearbysearch’ parts of the API with pd.iterrows().

The last part of class I learned about how to create visualizations based census data in combination with gmaps. It was good to go over how to install additional packages with conda. I also thought it was interesting to think about in the future how we will be working with multiple APIs for one project and possibly, merging data across multiple sources. Using gmaps was similar to creating visualizations using pandas/matplotlib with its own parameters when creating a heatmap. For the student activity, “Banking\_Deserts\_HeatMap”, I ran into an issue with calling the location for the heatmap and realized the pd.astype(float) from the solved version was the important fix.