Process Book

Authors: Eric Peterson & Caleb Ralphs

Overview and Motivation

When we first started to think about our project the first issue that came to mind was finding an accessible dataset on the web to utilize. Only then could we start to think about how we wanted to best represent the data. Caleb had previously worked with opendata.cityofnewyork.us/data data in the past and knew they had datasets available in different domains. We also had decided some sort of geospatial data would be best for this project as that is what we felt like we had the most experience in. After coming across the accident dataset it seemed to be the most appropriate to utilize for a geospatial application. The question then came down to how we could best represent the data and turn it into something that could be meaningful to a driver in NYC or perhaps some sort of city planner, civil engineer, or auto insurance provider.

Related Work

The first inspiration we had for this was the choropleth example in class which had to do with the unemployment rates by county. This visualization colored each county in the US on a scale according to the unemployment rate. The data that we used also had statistics by zip code so we thought to apply the same logic for this visualization in order to show the accident densities in each zip code, and furthermore, the likelihood of a future accident.

Questions

The question we were trying to answer is the likelihood of getting into an accident in different parts of NYC and representing this likelihood with the data available. As we moved through the project we shifted the question to be more specific to be what if I went from this location to another in NYC? What would be the percent chance that I would get in an accident through the course of that travel? How much risk is assumed when I go from point A to point B?

Data

You can find how we used the data in notebook.ipynb or script.py.

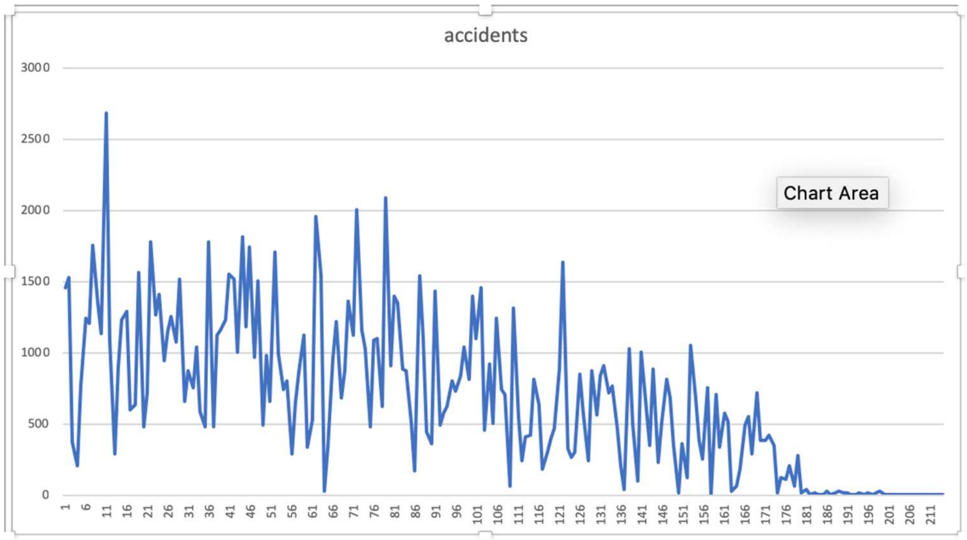
Accident Reports: The first dataset that we used was the accident reports in NYC in the year 2017. This data can be found here: https://data.cityofnewyork.us/Public-Safety/NYPD-Motor-Vehicle-Collisions/h9gi-nx95. We chose to only use one year of data for computational time constraint purposes. Once all the data was loaded into a jupyter notebook cell, we just aggregated the data into a two column data frame of zip codes and accidents.

Streets: The second dataset that we used was the streets in NYC, including the polylines for the actual roads and information on where the road starts and ends. This data can be found here: https://data.cityofnewyork.us/City-Government/NYC-Street-Centerline-CSCL-/exjm-f27b/data. The polylines for the streets and the locations of the start and end allowed us to aggregate all of the lengths of the roadways into each zip code. This allowed us to calculate the accident densities for each zip code (i.e how many accidents are reported for each mile of roadway in a given zip code).

Zip Code Shapes: The data set that we used to pull all the data together was the following: https://github.com/fedhere/PUI2015_EC/blob/master/mam1612_EC/nyc-zip-code-tabulation-areas-polygons.geojson. This was a geojson file of all the outlines for the zipcodes in NYC. We intersected these polygons by zip code with the associated road lengths per zip code and accidents in the zip codes. We then injected the accident density scoring into the geojson file to be used in our script.js to display the geojson as a choropleth.

Exploratory Data Analysis:

It was difficult to view the data in any sort of meaningful way until the data was run through the previously mentioned python script. Having a list of tens of thousands accidents in an excel document was almost impossible to look at or gain any insight from. After converting the original .csv to one with the amount of accidents per zip code. From there we were able to view a simple line graph in excel to see how much fluctuation there was in accidents per zip code. This original graphic can be seen below:

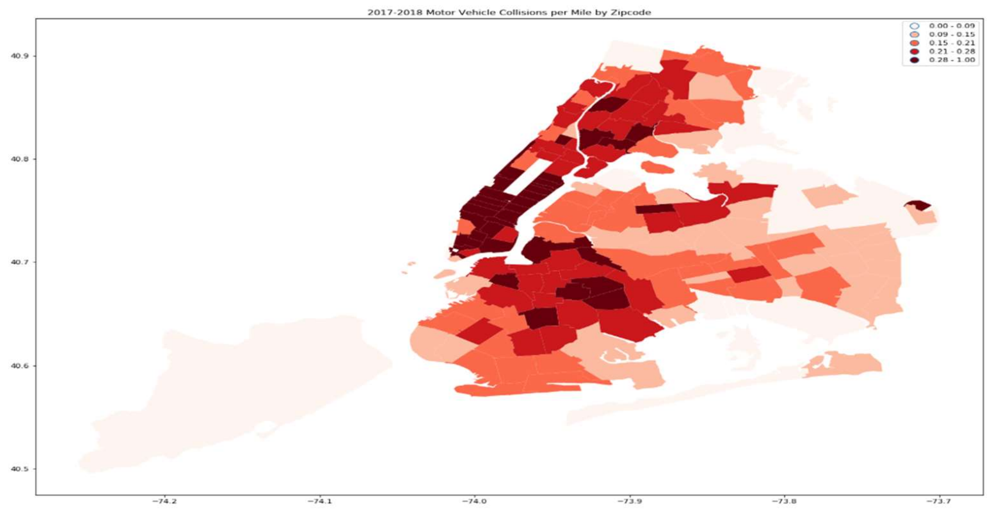


This showed us that there truly were large fluctuations by zip code and moving forward with implementing a choropleth map could be useful as there was enough variation. This is also where we noticed that there were zip codes with little to no accidents so we had to go back into the data to determine why this was and make the adjustments we mentioned previously. It turned out this was because it was mailing codes containing a single building or Central Park. As such there was no data to be reported for accidents and we should most likely exclude these outliers from our visualization to provide a more useful story for the user.

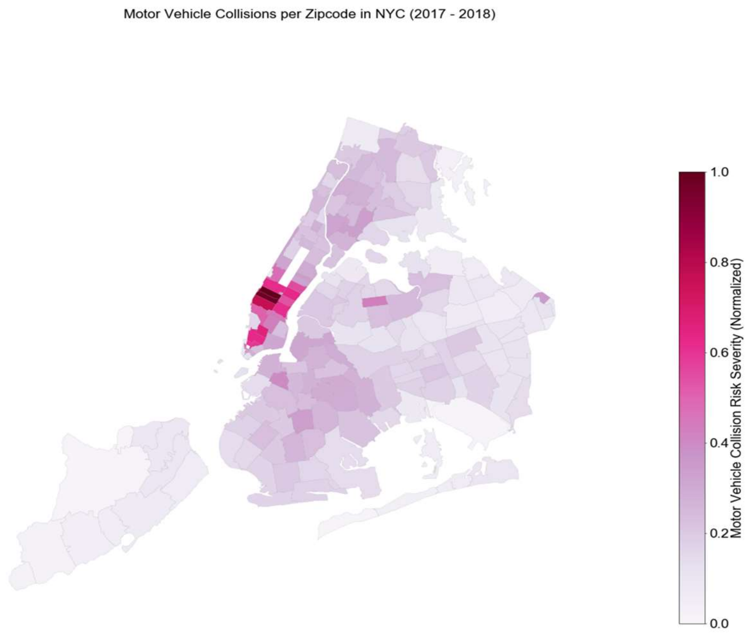
Design Evolution

As we mentioned previously were pretty set on utilizing a choropleth map because this was something, we had experience with and the channels which it offers allows a user to better visualize attributes of geographic regions. Then the first questions came down to how to properly implement a color scale to implement. Should we normalize the data and have a ramp function have the data scaled from 0 percent to 100 percent on some sort of accident score rating. This question came on the basis of where the zero scale needs to be. As we talked about previously, we wanted the user to understand the risk of an accident so had to make a design decision on what the baseline was for “no” risk. While this is difficult to really do, we decided to make the risk relative to only the dataset. This meant setting the lowest number of accidents in a zip code to be a zero and the highest number to be a 100 percent risk. To accomplish this we had to give each of the zip codes a risk score between 0 and 100.

As we started to think more about the data, we began to think about how there could be holes in the story we were presenting to a user. We looked at how different areas might have a higher score due to simply having more roadways not necessarily any type of inherent risk. This led to thinking about how we could scale the visualization according to the number of roadways within. To do this we implemented a new portion of code in the python script which would take the miles of roadway within that zip code and calculate the accidents per mile of roadway in the zip code. Then using this we calculated the same normalized number for risk of that zip code from 0 to 100. This gave us an “accident score” as we refer to it in the visualization. Using this score, we had to make design decisions with respect to the way scores would correlate to color. First, we tried an implementation which used a range of values for each color in 5 different bins. This can be seen in the graphic below.

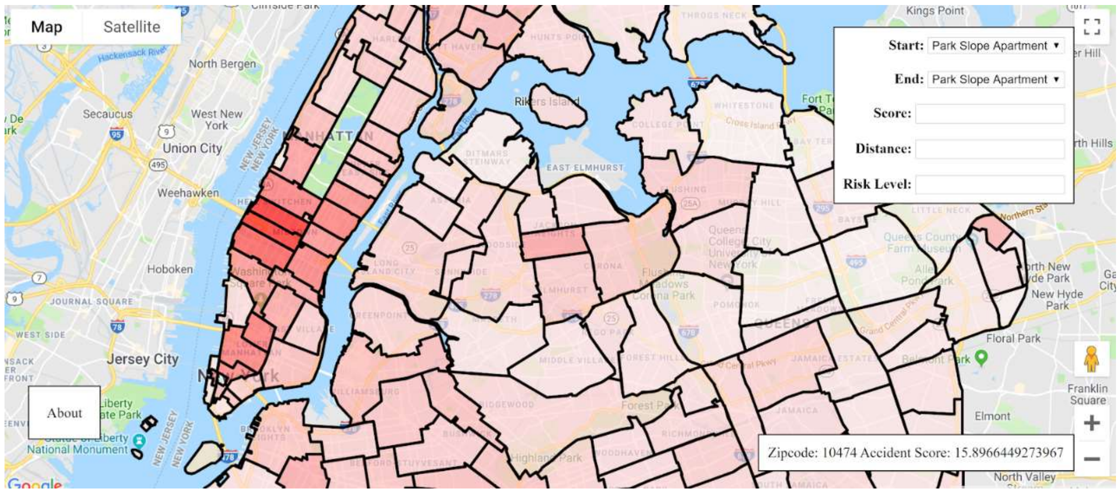


We then felt that this wasn't accurately depicting the ranges of data properly and was categorizing the zip codes rather than creating a scale out of them. Thus, we tried to implement a gradient scale based on accident score which can be seen in the graphic below.


Motor Vehicle Collisions per Zipcode in NYC (2017 - 2018)
1.0
0.8
0.6
0.4
0.2
0.0
Motor Vehicle Collision Risk Severity (Normalized)

From there, once we had the initial choropleth done, we started to think there could be more we could be doing. This is where the idea came about to give the user the ability to map between two different locations and understand the risk. Based on knowledge gained from the Software Engineering class we had both taken we decided it would be useful to provide a drop down menu for a user to select popular locations and then draw a line between those locations and come up with a way of allowing the user to understand risk on a given route rather than just per zip code.

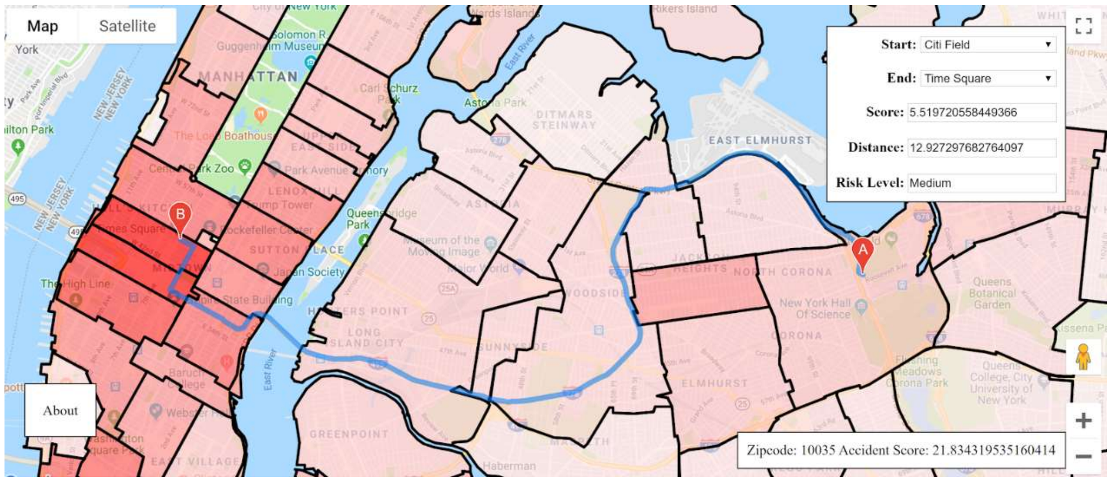
To do this we decided to utilize the API available for Google maps and overlay the choropleth we had created on top of it. This allowed the user to see zip code dangers, specific areas in their drive which may be riskier as well as that overall calculated risk. Please see the image below for our Google's Maps API with choropleth overlay. We did not think it was really necessary to use d3, since we were using Google's Maps API. The API handled a lot of the functionality that was necessary, like loading in the geojson polygons and handling all the functionality related to routing a trip.



Start: Park Slope Apartment
End: Park Slope Apartment
Score:
Distance:
Risk Level:

Zipcode: 10474 Accident Score: 15.8966449273967

From here we had to decide how to calculate a route and how to use that to output a calculated risk. The polyline for the route is created using Google's Maps API's built function DirectionService, and the DirectionRender function then overlays the route on the map. That polyline, the route information, is then used in order to calculate the length of time spent in each zip code and the accident score per zip code traveled is weighted with distance traveled and combined to give a total accident score for that trip. Please see image below for calculated route example.



Start: Citi Field
End: Time Square
Score: 5.519720558449366
Distance: 12.927297682764097
Risk Level: Medium

Zipcode: 10035 Accident Score: 21.834319535160414

This concluded our development phase and we will now address functionality and how to interact and use the visualization.

Implementation

The intent of this visualization is to allow a user to identify risk probabilities for motor vehicle accidents. The gradient red scale overlaid on the Google Map is representative of the accident score mentioned above for the given zip code. The user can pick starting and ending locations to calculate the risk associated with that specific route. The key design features can be seen in the design section above. Additionally, when the user hovers over any given zip code, the box in the bottom left changes to reflect the actual zip code for that region and the accident rating for that region.

Evaluation

By using this visualization, we learned about the different hot spots for car accidents to occur in the city. This could additionally offer insight for people be able to dictate the route with which they take and areas to avoid. Just as you have the ability to tell your GPS to avoid tolls, this allows an individual to decide if they want to avoid an geographic area using a data visualization implemented a popular user interface for navigation. We answered the questions of risk associated by driving both in certain zip codes and certain areas by implementing an easy to read choropleth implementation with the Google's Maps API. Overall, we think our visualization is very effective in displaying the intended information to users. However, we think there could be room for improvement based on data manipulation and backend functionality.

One thing that could be done to improve this further is to categorize every single accident by the severity or root cause and giving different values to say fender benders to accidents resulting in hospitalizations. This would allow for more accurate risk calculations rather than just risk of some sort of car accident.

Another improvement for further work could be the ability to plug in addresses rather than just common locations in the city so users truly could see how risky their daily commute is. We ended up not implementing this because, we were trying to work with the Snap to Map functionality for Google Maps, but it was glitchy when trying to drop points on a road. For presentation sake, we chose to just show common locations, but letting the user choose the start and end would be more useful.

The granularity of the risk density calculation could also be pushed down to the street level. If you were to implement this model/visualization in another region where the zip codes were not so small, then the model would not be very representative of the risk in an area. In one zip code you could have thousands of intersections, some more accident dense than the others. There are published papers around tagging GPS locations with the closest road polylines on a map, but this would require extensive future research to implement on our side. The street level granularity would make the risk calculations much more accurate.

Lastly, this was limited by just the accidents in NYC and as such the baseline for riskiness is just the lowest zip code provided in that dataset. However, if we could gather the information for all of the US we could create a map showing areas of riskiness with the entire country as the range for the scale. Overall, the scalability of this visualization could be easily implemented if the data were to become available. With these improvements in mind we do believe that we were able to accomplish what we set out to do with visualization and are happy with the result.

References

- https://developers.google.com/maps/documentation/javascript/tutorial
- https://stackoverflow.com/questions/38491370/how-to-add-geojsonmultilinestring-layer-to-a-google-map
- https://stackoverflow.com/questions/5623838/rgb-to-hex-and-hex-to-rgb
- https://stackoverflow.com/questions/22521982/check-if-point-inside-a-polygon