INFO 4310 HW4

Natasha Armbrust (nka8), Abhimanyu Gupta (aag245)

Dataset

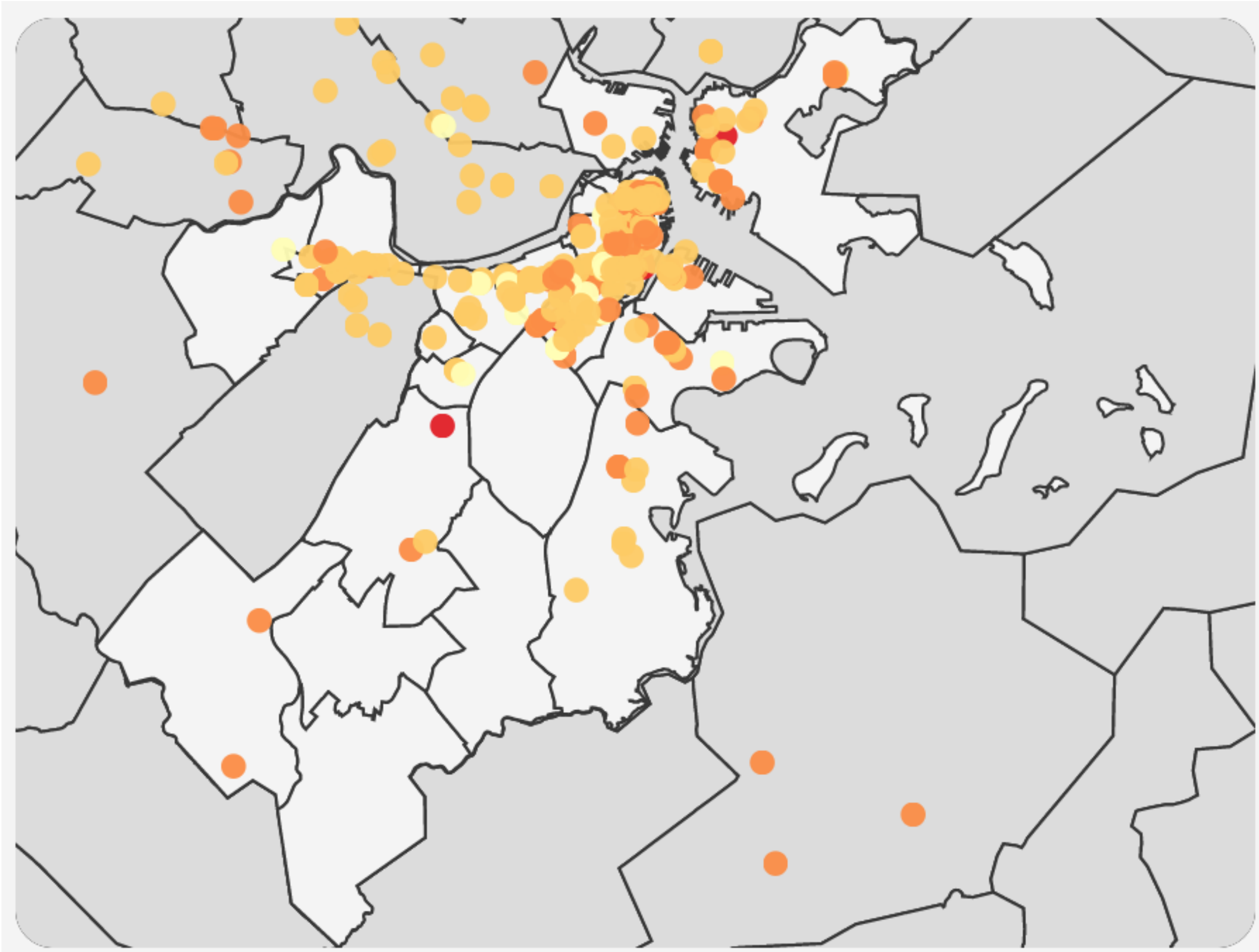
For this assignment, we chose to use the Yelp dataset from Boston. The dataset includes fields such as name, image, review count, rating, snippet text, location, neighborhood, and search category. The dataset did needed post-processing of the location field to determine the correct longitude and latitude coordinates (we did this before Professor Rzeszotarski posted a new fixed dataset). We also needed to post-process the categories field to determine the correct categories (ex: coffee, sandwich, italian) associated with each eatery. Before we did any post-processing, we first characterized the behavior of a user trying to make an informed decision about this dataset. We assumed that a user would want to sift through this dataset to determine an appropriate place to eat depending on their specific criteria. Below we outline the interactions we wanted to provide for users:

- Find an appropriate eatery
- Filter on specific criteria (ex: only italian restaurants in the North End with at least a 4 star rating)
- Easily save potential eateries and compare these to new eateries being explored
- Discover new eateries spontaneously

We realized that with the time allotted we could not make the equivalent complexity of Yelp, however we found creative ways to satisfy the user affordances listed above with a novel interface.

Potential Challenges

Potential challenges in interacting with the data come mainly from the high density of data we have. We only have a subset of Yelp's data (around 300 eateries), but this is still enough to crowd a visualization and make it hard for the user to explore the dataset. On a static map, if we plot all the eateries we get the crowded image shown below. Clearly we needed to give users the ability to zoom into certain locations. Allowing users to filter based off of criteria also cuts down the size of the dataset and makes it easier to explore.



All Eateries Plotted on a Map of Boston

Another issue that arises from the high density of data is the idea of "where to start?". Often times users don't know quite what they want when exploring eateries and are open to a wide variety of options. Over 300 eateries can be hard to sift through if you are a general user who is fine with "any type of restaurant in inner Boston." A potential challenge is discovering eateries spontaneously.

The information provided for each eatery in the dataset is also quite limited. For example, the dataset only provides a snippet of text from one review. A user might have a hard time making a decision about eateries given just the dataset. However, the dataset does provide a link to that eatery on Yelp. We wanted to allow users to get to the Yelp detail view easily from ur visualization.

Finally, the dataset contains only 300 eateries. When filtering for specific criteria, the dataset can be extremely reduced. For example, there is only one eatery in the neighborhood Coolidge Corner. Due to the limited dataset, users might find it hard to find eateries that meet their specific criteria.

Interactions

Filter + Dynamic Query Systems

We wanted to give the determined user who wanted "only italian restaurants in the North End with at least a 4 star rating" the ability to use our tool as well as the general user who is fine with "any type of restaurant in inner Boston." Thus, the first affordance we give users is to filter based on criteria. If no filters are chosen for a field, it does not affect the query. We wanted to make this filtering obvious since it helps the user explore the dataset by reducing the number of datapoints. Thus, our final visualization features our filtering UI front and center in a navigation bar.

UI For Filtering

We determined that the relevant metrics to filter by were minimum rating, minimum review count, categories, and neighborhoods. Both rating and review count are number inputs that have a min and max bound. The category and neighborhood field are dropdown menus with the ability to select one or more neighborhoods. We created placeholder text with "pick one or more ..." because we wanted users to know multiple selections were possible. Of the other fields given in the dataset, we picked these metrics to filter on because they allowed for exploration and reasonable criteria for a dynamic query. We chose not to allow a user to input a specific eatery name because we wanted to encourage exploring.

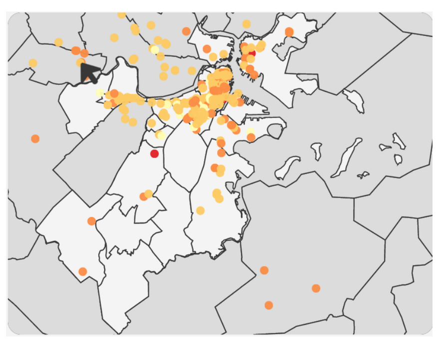
Upon the user picking filtering metrics and clicking filter, we create a dynamic query of our dataset to display only relevant results.

Pan & Zoom

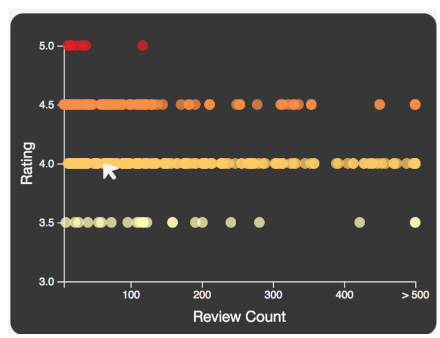
As described above, the amount of data points on the map of Boston crowds the exploration; the data points overlap each other in many areas. To allow a user to successfully explore the dataset geographically, we created a pan and zoom feature on our map of Boston. Upon clicking, the map will center in the area clicked and zoom in. The circles plotted as each eatery reduce in size so the map becomes easier to explore. Upon hover, a tooltip shows the eatery description and upon click the eatery is selected. A selected eatery is shown in a accessory view and highlighted on both the map and graph.

Zoom & Pan UI

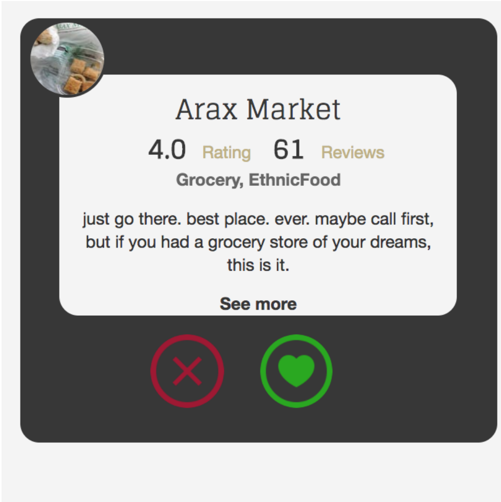
Selected Eatery in Zoom In



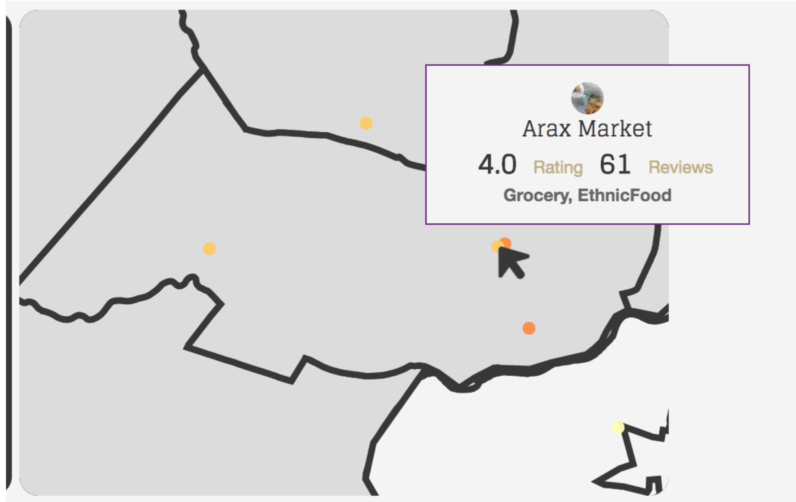
Selected Eatery in Zoom Out



Selected Eatery on Graph



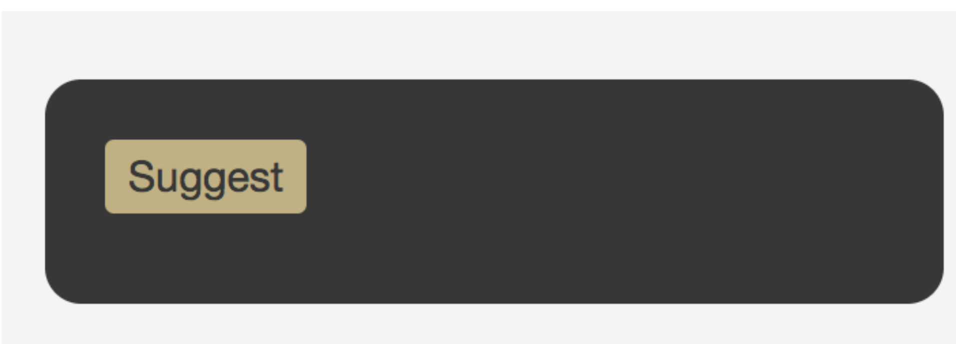
Accessory View for Selected Eatery

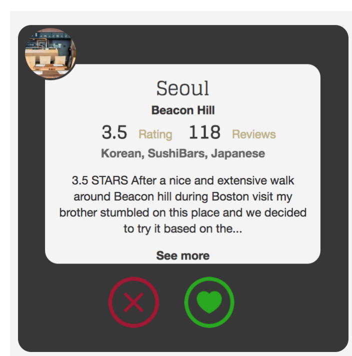


Tooltip UI

Recommender System

We also wanted to provide users with the ability to spontaneously discover new eateries. Our original idea was to create a "Tinder for food." To explain, rather than giving users the choice on which eateries to explore, we wanted to show a potential eatery and have the user like or unlike that eatery. However, after exploration of the dataset and user testing, we realized users wanted the ability to compare eateries, which meant displaying more data than just one eatery at a time. Thus, we kept the original idea of suggesting a potential eatery to users and incorporated allowing the users to explore the dataset by clicking around on the graph / map. The eateries we suggest meet the filtering criteria. We would have liked to create a robust recommendation system based off prior eatery likes to suggest new eateries. However, given time constraints we were only able to create a randomized suggestion that met the filtering criteria. Random or not, the suggestion feature still allows user to discover eateries spontaneously and reduces the information overload of >300 eateries.

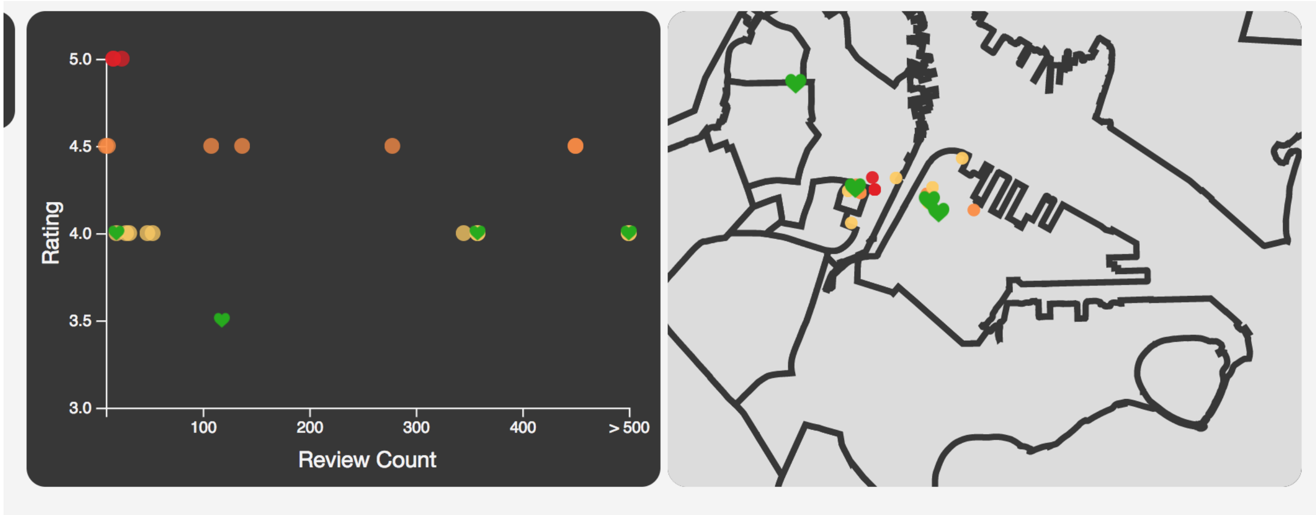




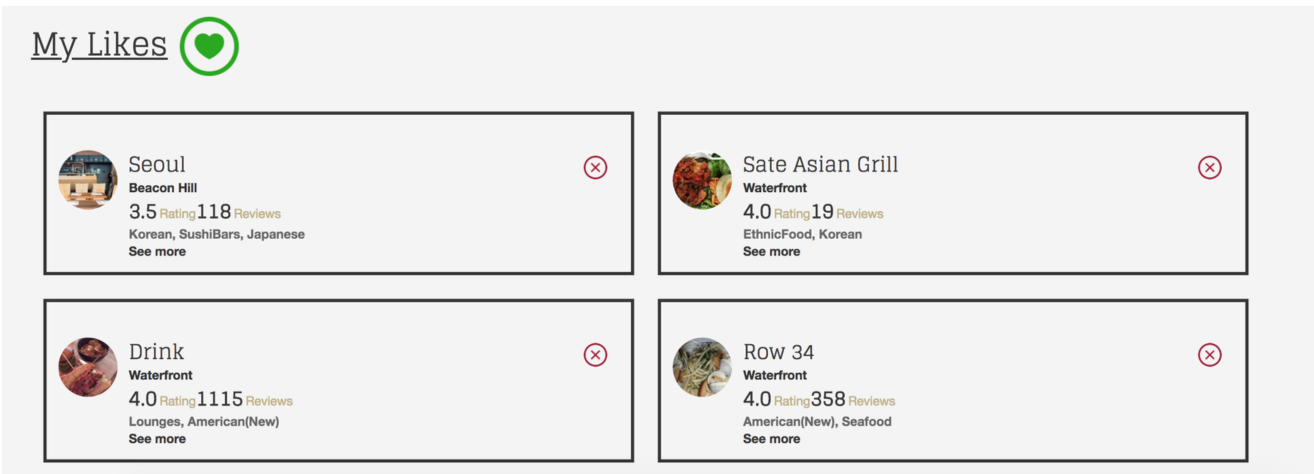
Suggestion UI

Comparing Potential Eateries ("My Likes")

From our original idea of "Tinder for food" we wanted to satisfy the user need of storing potential eatery options and comparing those "likes" to new eateries being explored. To do this we created a "liking" system that displays the liked eateries on both the graph, map, and in a list below. By highligning a users liked eateries, we allow the user to compare potential eateries they were interested in with new eateries. All the liked eateries are displayed on the map and graph even if they do not meet the current filtering criteria.



Liked Eateries Featured on Graph and Map

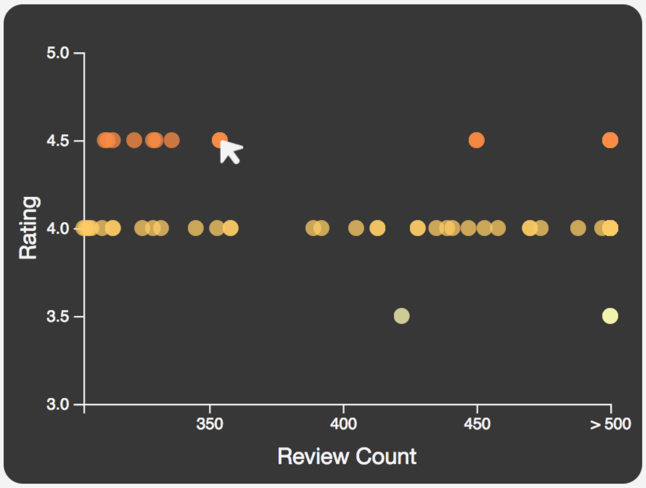


Liked Eateries Displayed Together

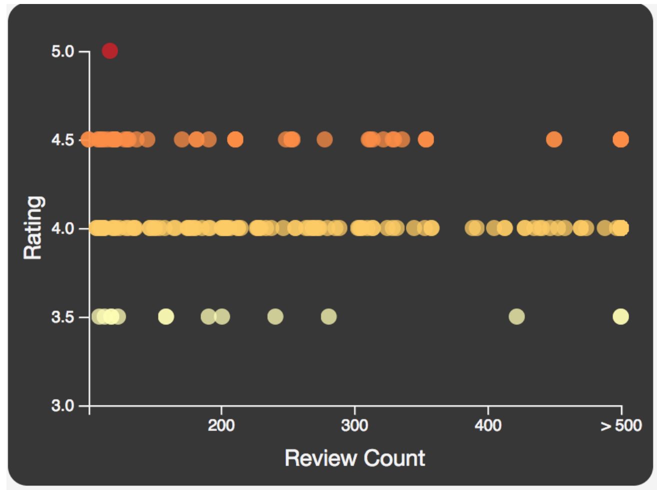
Final Visualization

Our final visualization is a tool to help users discover eateries in Boston that fit their criteria. We provide a navigation bar that allows the user to filter the data by various metrics such as minimum rating, minimum review count, categories, and neighborhoods. Upon clicking filter, we display the eateries in the dataset that match the criteria. We provide two unique displays of the eateries: a graph of rating by review count and a map with eateries colored by rating.

We included a graph of review count versus rating. In our personal Yelp searching experiences we noticed the lack of easily comparing eateries review counts and ratings. The review count is more of a hidden number in many of the current food finding websites. We wanted to unhide review count and display it against rating to show a novel view of eateries to the user. Each of the points on the graph corresponds to an eatery. On hover, the eatery is displayed in a tooltip and on selection a more detailed display of the eatery is shown (including the ability to "like" the eatery). The color of the eatery on the graph is a scale that corresponds to the rating. We used a d3 yellow red color scale to show the rating levels. Our dataset only had 4 different ratings within it so our color scale has four distinct colors. The points on the graph have < 1 opacity so that multiple points that overlap can be shown. To reduce the overlapping points, we resize the x-axis upon filtering to be the extent of the review count in the filtered data. Any review count > 500 we display as 500 or greater on the graph. This allowed for more visibility of overlapping points. Our review count filtering input only allows a maximum of 500.



Resizing x-axis



Resizing x-axis

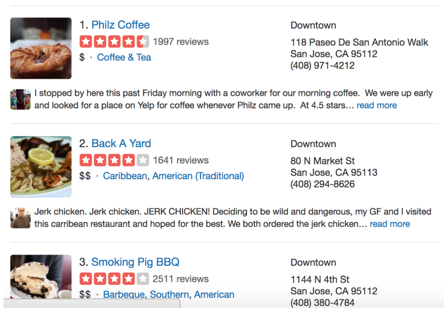
We included a map of Boston with plotted eateries. We chose to include this map because many users choose eateries based on location and are used to looking at eateries on a geographical display. We personally believe that our visualization would be lacking if we did not have a geographic component to our visualization. We plot each eatery on the map with color corresponding to rating. We included the coloring based on rating because in Yelp, Google, and other food finding apps, rating is hidden when looking at the geographical component. This causes users to click on many eateries with lower ratings than desired because they are geographically close. We allow users to pan and zoom on the map. Upon hover each eatery is shown in more detail with a tooltip and upon click the eatery is shown in an accessory view and highlighted on both the map and graph with a selector icon. Only the eateries that meet the filtering criteria are plotted on the map. When a user selects certain neighborhoods on the map, those neighborhoods are highlighted. The default white highlight of the map is considered “inner” Boston area.

We include a suggest feature which allows for spontaneous discovery by the user. Upon clicking suggest, we suggest an eatery to users (that fits the current filtered criteria) and highlight it on the graph and map.

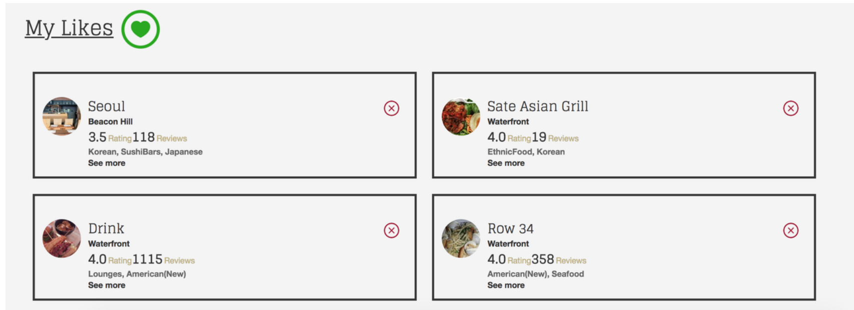
We include a liking feature which allows users to save eateries for later. A liked eatery is plotted on both the map and graph with a green heart icon even if it does not fit the current search criteria (given it fits in the x-axis for the graph).

Trade-offs

Our biggest issue with the project was the alloted time. We felt we did not have enough time to fully create the polished assignment we were hoping for. Due to the limited time, we had to make design decisions that came with tradeoffs. First off, we went for exploration on a graph and map versus a list view. Many food finding websites will display a list of eateries that meet the criteria. The image shown below is an example of Yelp’s food finding list. We decided not to show a huge list of eateries. The large list becomes overwhelming to a user, and we did not have a recommendation system to rank the eateries from best match to worst match. A tradeoff for not including a list is not showing users an easy way to see the descriptions and details of eateries side by side. We alleviated this tradeoff by including an accessory view that shows an individual eatery in detail. Upon liking the eatery, it shows up in a “My Likes” list below. Since we did not have a recommendation system, we believed only showing a list of eateries that the user was interested in was a better choice for our visualization.



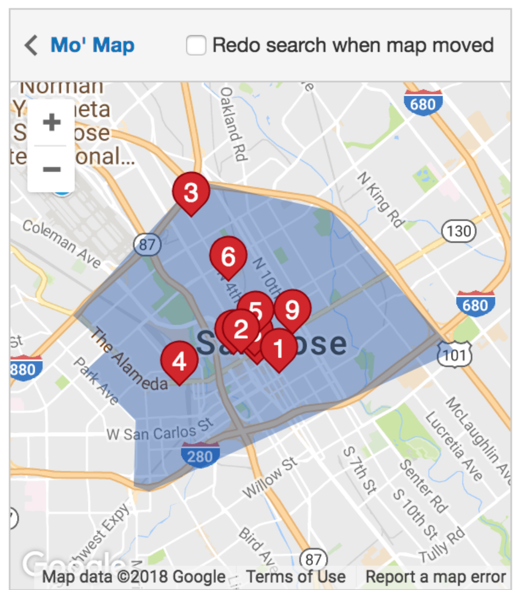
Yelp’s List of Eateries



Our List of Eateries (Only Includes Liked Eateries)

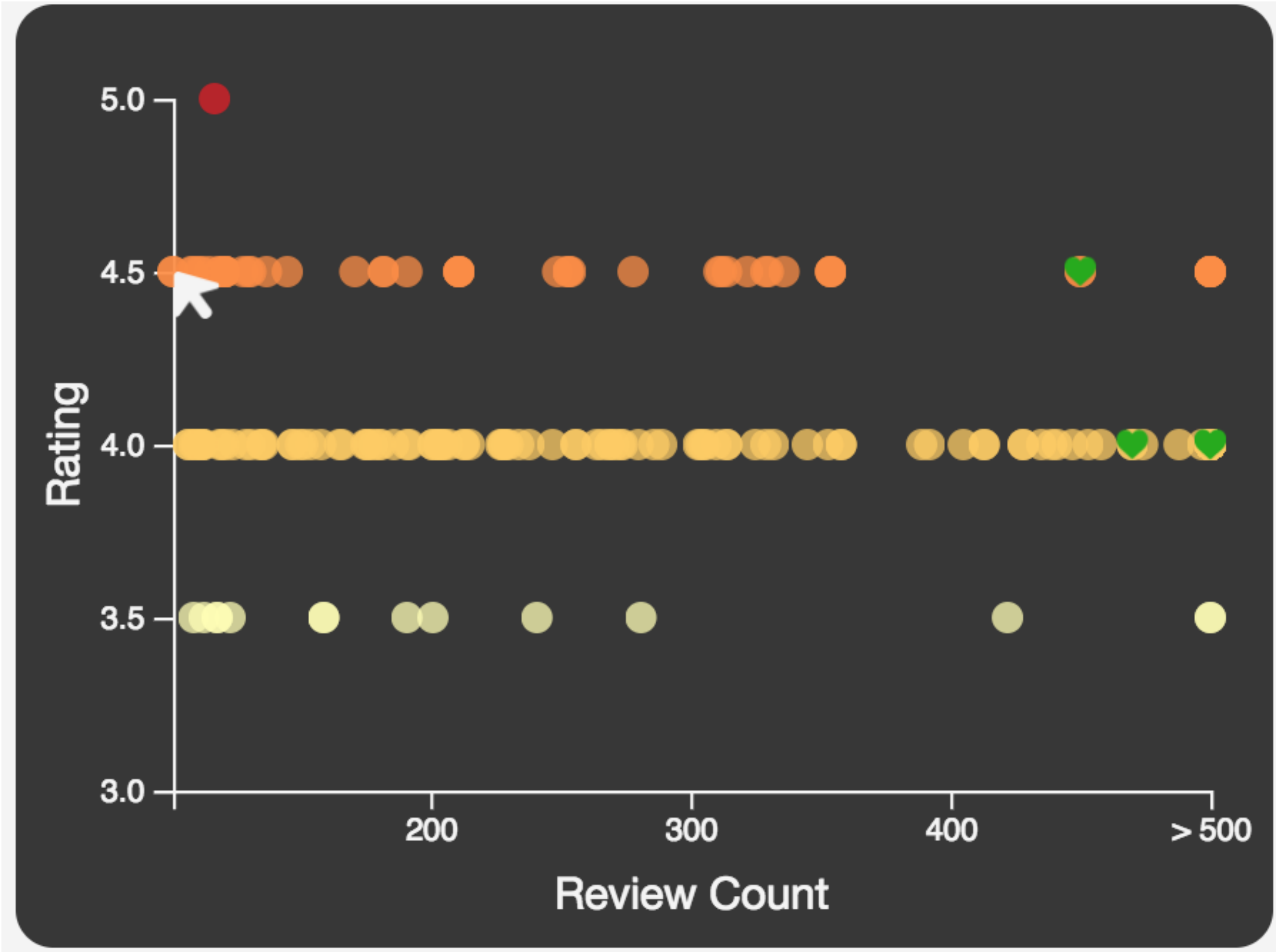
Another tradeoff we faced was the high density of points. We tried to alleviate this be creating a pan and zoom feature on the map. However, given limited time, we were not able to perfect the pan and zoom feature interaction. Some points still remain hidden upon zoom. Given more time we would have alleviated this by allowing different zoom levels and allowing the user to hide some eateries on the map. We also thought of allowing the user to toggle how many eateries they wanted displayed on the map and graph (ex: display only highest rated eateries, display all eateries, etc.). We noticed even Yelp has trouble displaying eateries geographically. They alleviate this issue with a multiple level zoom feature.

Hidden Points on Zoom



Yelp’s Eatery Map

We experienced the same issue with highly dense eateries on our graph. To alleviate overlapping points we reduced the opacity of the circles and resized the x-axis to fit the dataset. Given more time, we could also have allowed for a zoom in feature on the graph.



Hidden Points on Graph

We would have liked to create more of a discovery element to our visualization. Currently, we have a "Suggest" button that will give the user a suggestion based off their previous likes. However, we did not have enough time to implement the recommendation feature. Because of that, our current suggestion is a random eatery that fits the filtered criteria. Given more time, we would have created a basic recommendation system. This basic recommendation system would also have allowed us to only display highly ranked eateries to the user. This would have helped alleviate the overcrowding on the map and graph.

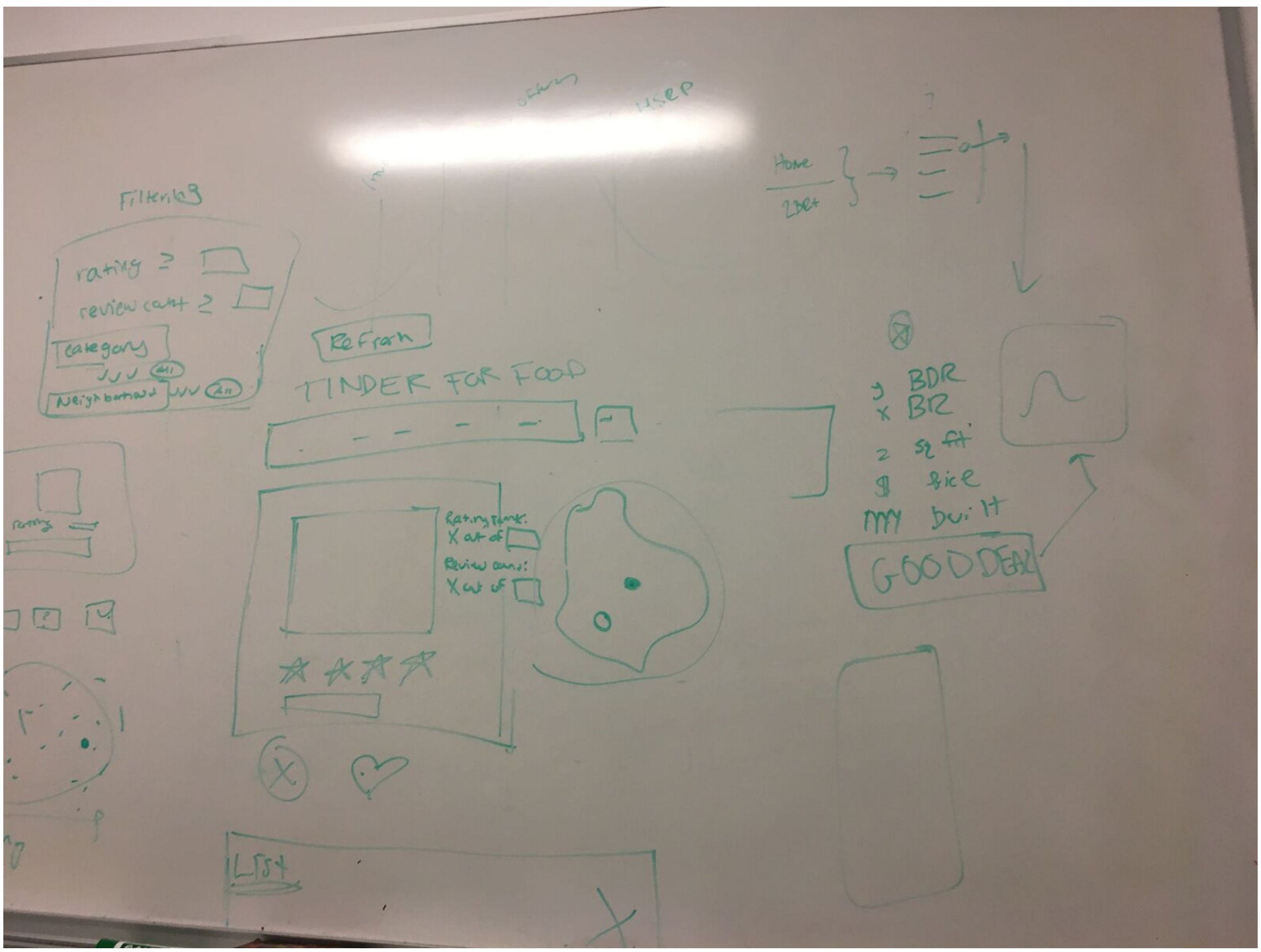
Finally, we struggled with plotting the points on a map of Boston in d3. Unfortunately, we could not find a street level geoJson of Boston. Thus, the level of detail is very minimal in the map. We would have liked to have shown street level views of the data. Given more time, we probably would have switched to Google Map's API instead of using a d3 map of Boston.

- Briefly outline the development process of your tool. Explain how your visualization/interactions changed between design and final implementation. Comment on any trade-offs or design choices you had to make while developing.

Development

We used javascript, d3, html, and css to develop our tool. Unfortunately, the development took a significant amount of time and some of the nice final touches (ex: animations, UI display, etc.) we were not able to get to in time. We first post-processed our dataset by using a regex on the fields in the dataset that required parsing (such as categories and location). We then worked on filtering logic by dynamically creating a selectpicker based off the neighborhoods and categories our dataset contained. We used bootstrap and jQuery to help with the UI development.

Our initial idea for the visualization was to create a "Tinder for food." This meant discovering eateries one at a time. As explained above, rather than giving users the choice on which eateries to explore, we wanted to show a potential eatery and have the user like or unlike that eatery. However, after exploration of the dataset and user testing, we realized users wanted the ability to compare eateries, which meant displaying more data than just one eatery at a time. We realized if we gave users the ability to compare the current suggested eatery amongst others, than the user would expect the interaction of clicking on the other possible eateries. Therefore, we ended up moving away from a "Tinder for food" idea and moving more towards our final visualization. Below is an image of our initial storyboard.



Initial Storyboard

In the initial storyboard we did not include a graph of rating versus review count. Instead, we thought of including text for the rating and review count rank. For example, "Rating: 5 out of 500 eateries that match this criteria." However, as we were developing the tool we realized we were lacking visualizations. In addition, user testing showed that when we displayed "rating: 5 out of 500 eateries that match this criteria," a user expected to be able to see the eateries rated higher. Our current text display did not give users the ability to compare eateries, so we added a graph of rating versus review count.

Providing a graph of rating versus review count with all the eateries that met a certain criteria led us to also plot these points on the map. Our original idea was to only plot "liked" eateries and the current selected eatery on the graph. However, once we added the graph of rating versus review count, the map became unintuitive. In addition, user testing showed that users wanted to select eateries geographically as well (not just by rating and review count). This led to us plotting all the eateries on the map as well. Unfortunately, the eateries overcrowded the map. We realized we had to add in a pan and zoom feature to make the map legible for users.

Developing the UI took much longer than expected. Due to the time sink, our visualizations were less polished than planned. We would have liked to create animations for points on the graph and map upon zooming and/or filtering.

Work Breakdown

Brainstorming + Storyboarding
Both Abhi and Natasha brainstormed and storyboarded the original idea. This was the most creative part of the project and took the most mental energy. Roughly 4 hours was spent brainstorming and storyboarding.

Initial Data Exploration
Natasha explored the initial dataset and post-processed the dataset. This was relatively straightforward and took roughly 2 hours.

Visualizations
Natasha created the basic UI for filtering and filtering logic. The UI work took much longer than expected, but the filtering logic was pretty straightforward. This took roughly 8 hours. Abhi created the graph and plotting features. This took roughly 6 hours. Natasha created the geolocational visualization. This ended up taking more time due to the error in the dataset (long, lat were incorrect at first). In addition, finding a geoJson of Boston was difficult and Natasha had to create her own by overlapping two different geoJsons. This took roughly 10 hours. Both Abhi and Natasha worked on creating the liking feature and final touches. Since this required going back to the drawing board and fixing up final interactions this took a large chunk of the time > 10 hours.

Report
Both Abhi and Natasha contributed to writing the final report. This took roughly 5 hours.