Christian Lee
Email: clee@cs
GitHub: clee11

**Problem/solution**
The main problem is largely started/solved by revminer’s basics. I want to make this better. I want to implement a better recommendation engine. Currently, you can click Agua Verde Café, and you will find a list of 5 “similar restaurants.” Now, however, I would like to be able to ‘+1’ a restaurant or a set of restaurants and have a list of recommended restaurants returned to me based on their similarity to the places I like. The difference is that I can now get similarity to a set of restaurants. And I need to do so on the fly.

Currently, the similarity calculations are done offline and stored prior to running the server. And like most similarities/categorizations for revminer, the current function is based on a KL divergence of all the places’ values. The two main obstacles I need to overcome are how to do this fast/online and to figure out a better heuristic. It is easy to determine KL divergence from a single place, but difficult for multiple places. It is not just determining distance because +1-ing a fancy Italian restaurant and mcdonalds and chinese dim sum are likely to have completely different values.

**Artifacts**
I want this done as closely related with the current revminer project as possible, so I would add on to the javascript server and client (both web and mobile), senchatouch, mongodb, etc. I will probably have some Java code run the similarity calculations, but I will likely learn and port this to python to keep the format similar with the rest of revminer’s scraping/calculations. The data would result in an object as simple as

```
Place1: Similar1, similar2, similar3, similar4, similar5
Place2: Similar1, similar2, similar3, similar4, similar5
```

I want this to work as easily and user friendly as possible both on HTML5 and mobile. And of course I will include source code, user/developer documentation, and app package for all of this.

**Methods**
- Node.js, mongodb, python for offline calculations
- Joyent→AWS→random page off of my parent’s server→anything that works
- Data is just the attributes and values for each restaurant that has already been stored
- I do have to query it realtime because if the user +1s a new place, the recommendation must change. But I imagine I can save the user’s old data and store that and only make small changes. I imagine I’d have an offline graph grouping similar restaurants into smaller chunks so that I don’t have to iterate through ALL restaurants when someone +1’s a new place.
- No multiple sources
• I need to find the best way to find similar attributes. I assume there are preexisting machine learning or information retrieval algorithms to help me out, but I haven't looked. I know there are lots of ways to train/test data as long as you do not train on the exact same data. I will likely try a smaller chunk of data and simpler data off of some machine learning or AI book to make sure my algorithm works, then port this to revminer’s data

• I think I’m going to copy Google’s +1 interface. It’s just a small button on the results or match page, so it’d easily fit on a small mobile screen.

Split up work
I’m working by myself

Milestones
Jan 26 – find a good heuristic. Have it working offline. Figure out a clustering of restaurants so when we do calculate online, we don’t have to search ALL restaurants.
Feb 16 – have it working on HTML5 with pretty UI
Mar 6 – port to mobile, cleanup
Document everything along the way. Learn the mobile infrastructure along the way.

Success
General: I can ask food connoisseurs (friends) about what they think of the similarity. I know a lot of restaurants, so if it’s similar prices, same type of food, maybe similar area. I can probably judge pretty well if it works

Statistical: I can pull a random example of a set of +1’d restaurants and the listed recommendations. I can show the distribution of their key attributes/values and show how the top recommendation is better (based on my heuristic) than the 10th

Other
No need for extra data/scraping
My new data is a simple JSON object that will most likely be (partially) thrown in the database for faster querying. Old data is already in the database of restaurants and its values/attributes/frequencies
I will manage code through Git