Senior Project Paper

My senior project is a proof-of-concept system for providing targeted recommendations by dividing users into subsets referred to as wavelengths. The project uses an instance of the Node.JS framework hosted on Heroku and uses the Heroku-provided Postgres as its datastore.

This project has three measurable outcomes.

1. Can the system provide targeted recommendations?
2. Can the system provide estimates of user enjoyment on specific content?
3. Is there a functioning website to provide these services?

All three outcomes have been realized. The recommendations page provides a registered user with targeted recommendations. Each content item has an associated webpage that provides the general consensus rating, as well as an enjoyment estimate for the user.

The base of this project is the website. This site is active and available for users to interact with and examine. (https://cit490-senior-project.herokuapp.com) Without logging in, some recommendations can be generated, pulled from the aggregate of all existing user ratings. Ratings are registered in a simple binary fashion, with a logged-in user selecting “thumbs up” or a “thumbs down,” indicating approval or disapproval, on the content items presented.

When a user account is first created, they are not assigned a wavelength. After rating their first content, the system assigns them the first wavelength that aligns with the rating they performed. At this point, if we compare the user’s ratings to the wavelength, 100% of their choices match the wavelength. This comparison is called sync.

When any of a wavelength’s users rate any content item, the wavelength’s intensities change to reflect the additional data, growing toward an intensity of 0 if none of the users like it, and 1 if the users approve of it. A user’s sync is recalculated periodically. If a user has rated enough items counter to the concordance of the wave, they will fall below the sync threshold, a number set for each wavelength to determine how accurate the wave is trying to be.

Falling below this value triggers the system to pull the user out of this wavelength and check their sync with other wavelengths. That user is then placed in the wavelength that best matches their like/dislike profile. If no wavelength fits their needs, meaning that they are under the sync threshold in all established waves, a new wavelength is created using the rating profiles of this user as the base data, so they once again match at 100% sync.

This process repeats as they rate more and more content, their wavelength sync wavering up and down as they move from wavelength to wavelength, until all users settle on wavelengths that fit them perfectly. In these ideal wavelengths, all of the subsets of users should think in a similar way. At least, in terms of what content they enjoy. Recommendations drawn from these subset of users should, in theory, be more likely to depict something a user in a wavelength would enjoy when compared to the results pulled from simply averaging how all users have rated a content item.

In my proof-of-concept, I am able to provide recommendations and estimates of enjoyment for user accounts. There are, however, limitations of what my small test can provide. Accurate recommendations require data. The more users, the more ratings, the more points of intersection between a user and their wavelength, the more accurate the system can be with its predictions of enjoyment. At this point in time, while my project demonstrates the functionality of the idea, there is insufficient detail to really be able to tell how useful of a system it will end up being.

One potential impediment that I've identified is the need for divisive content. My test users and I come from the same background, and our rating profiles look very similar. While this lets us build up strength in a wavelength, the more preferable occurrence is for a few items to be near 50% rating when looked at from all users, but for differing wavelengths to rate that content near 100% or 0%. A content rating that splits users into distinct dichotomies. Having a few pivotal pieces like that in the dataset would make it easy to sort users and demonstrate the utility of the system. At this time, my testing has not revealed any such items. Having and identifying these items could be very useful as a way to present new users with a quick way to get sorted into a viable wavelength.

I am very grateful for the opportunity I've had to work on this project this semester. I found myself wishing I could have worked on more projects like this across my school experience. The most powerful thing I've learned is how much can be done by a single person just putting

in 2 hours a day. I've built projects as part of classes before, but in those classes, they were guided by the rubric, where there was a steady hand from the course about what was required. But, as the lead designer, project manager, and gofer on my team of one, there was no explicit guidance. Instead of checking boxes for a grade, my design was based on the more important question of how well it performed whatever task I was working on.

My understanding of databases changed over the course of the project. My first self-taught forays into databases used them as essentially a fancy filing cabinet. But the more I began working with stored functions and procedures, I realized that I had misunderstood the relationship the Postgres server had with the Node server. Postgres wasn't a filing cabinet, it was a secretary. Functions could be offloaded to the database, where they can be written cleanly with solid programming practices, instead of a nested set of queries on the NodeJS side that was tricky to test and hard to implement. If I had realized this distinction from the beginning, I would have saved myself some headaches.

I also regret not having version control for interaction with the databases and the stored functions. Separating the production database from a test database would have given me more confidence in tests that may have been problematic if applied to the production database. In a related regret, all of the code that I wrote for the database is unavailable to the public, which is not ideal when that code is the star of the project. Having some version control options would be a way to have my concept available for easier dissemination and explanation.

Over the course of the semester, I've been confronted with a lot of things that I have not known how to do. This has been a great confidence builder for me, as no matter what I was faced with, I was able to do the research and overcome the challenge. I may not have been as fast as someone more skilled or practiced in the technology, but I was not stopped by ignorance. I feel this is an important attitude, as the last few semesters have shown me how much I don't know about Technology. My school experience has given me the barest taste of what is out there. There is no way for our degree to give us all the knowledge we need in our future careers. But the confidence in researching the unfamiliar and learning its secrets makes me feel prepared for finding a job. I know I won't have all the skills they're looking for immediately, but I know I'll be able to hit the ground running and learn what I need to be able to do the work.