**November 9 2017 11:30 am - 1 pm**

* Machine Learning Discussion
  + Scalability
    - It is not feasible to store a model for every user
    - Nor is it easy to train every user model
    - We plan to look into general models we can use for better accuracy and for storage size for scalability
  + Ensemble
    - We had many different suggestions for this
    - Currently we are looking into a couple different ideas
    - Only update the model every hour?
    - Only update the model when the user has been inactive for a certain period of time
    - Once we update, we have every portion of the ensemble train the model on the new inputs and update the queue/buffer of products
  + Queueing/Buffer
    - We know that it is impractical for performance purposes to have every portion of the ensemble run a prediction for every time a user requests a new product
    - The most likely idea to work is some sort of buffer of products that the user may like
    - Every time the model updates (detailed above) each machine learning algorithm will rate each remaining product in the database. They will rate the product based on the user’s likeness of liking that product. From this buffer or queue the server will select a product to serve to the user
    - We are also looking into storing the liked products in a queue and then they will be released based on the specifications listed above
  + Looking to get this set up relatively quickly
  + Once setup, we need to find a way to pass information between the machine learning algorithms and the server
  + In addition, we need to write some sort of ensemble program that can communicate between the different machine learning algorithms which are written in different languages
* We are still working on the mobile app looking to get that ready for a demo coming this next sprint presentation

**November 14 2017 10 am - 12 pm**

* Machine Learning
  + Scalability
    - General Models
      * We have found a way to remove the user model for every user and are attempting to implement a general model scheme
      * We plan on using a vector system
      * If users’ have similar vectors then we can group them together to reduce the number of user models we have to store
  + Ensemble
    - CNN
      * As we continued to improve this portion of the ensemble and we now have close to 66% correct on this learner.
      * Need to tie this into the ensemble along with the other learners
    - Naive Bayes
      * Coming along well. We have figured out the problem that was causing the learner to suggest all dislike
      * Now using two hidden layers, we are getting a 65% correct prediction rate
    - Decision Tree
      * We were also having a problem with this algorithm that made it predict 100% correct
      * We figured out this issue as well and we are now getting around 60% correct
  + Queuing/Buffer
    - Looking into using the queuing system to integrate between the server and the ensemble
* Front End
  + Need to add the search field on the liked products page
  + Need to make the website mobile friendly
  + Need to add the front end support for the general channel
* Back End
  + Added the back end support for search liked products
    - Need to integrate this into the regular liked products end point
  + Need to add the delete preference endpoint
  + Added the change preference endpoint into likes and dislikes
* Mobile App
  + Looking good so far
  + All added so far
    - Login
    - Register
    - Change password
  + Need to add the channel functionality at the very least

**November 16 2017 10 am - 12 pm**

* Machine Learning
  + Queuing
    - Still looking into adding the queue however there appeared to be a difference in versions which may not work for the current version of node we are running
  + Ensemble
    - Have a basic ensemble going for the Naive Bayes and Decision tree
    - Looking to add the CNN into the ensemble in a different way
    - Due to the length at which the CNN may run, we are having it preemptively rate products based off previous input to a database table
    - The ensemble will pull from that table for CNN and the CNN while run in the background to rate products accordingly for users
    - We are currently working to tie this all together on the back end
  + We plan on selecting a random product on the node backend, then sending it to the ensemble.
  + The ensemble will then feed it to all learners and we will collectively average out the what each learner returns
  + If the average is below a certain rating, we will feed it another random product until the ensemble returns with a high enough rating
  + Scalability
    - We have thought of a general solution for this.
    - We will be using weight vectors instead of user models, therefore it is simply storing a long vector for users and then combining them to be general models
    - For CNN, we are going to use clusters of users to reduce the size of training for every user. We will group users based on their preferences.
    - We will evaluate which cluster users go in based on their continual input
* Front End
  + Search field is now added and hooked up on the front end
  + Website appears to be good for mobile devices
  + Still working on general channel
* Back End
  + Added the delete preference endpoint
  + Added the change preference endpoint to the likes and dislikes for when a user changes preference
  + Added the random product for general channel to random product endpoint
* Mobile App
  + Channel function is currently working for the general channel
  + Working on adding all user channels
  + Like and Dislike work
  + Some bugs, but looking good so far