Brandon Sung

Part 1:

1. For Part1, I did the memory-based approach. In order to make predictions for each user in my corpus, I need to find other users in the corpus who are similar to my current user. I did all my coding in a single file python file named main.py.
    a. I used Pearson Correlation Coefficient Similarity to determine which users were similar to my current user. This is a version of vector space similarity, where each dimension is a user's rating for each dish.
    b. The problem is that none of the users have rated every dish in the corpus of dishes. Therefore there are null values when calculating the similarity. To get around this problem I came up with my own solution.
        i. First, I filtered my corpus of users to only contain users who had rated at least one dish that my current user had not yet rated
        ii. I calculated a similarity score for my current user(test_user) and each train data user(train_user) based off of the intersection of dish ratings they both had in common.
        iii. Then, I would multiply that similarity score by the ratio of the number of dishes they had in common over the number of dishes my current user has rated in total.
            1. This means that train data users who have rated more of the same dishes that my current user rated would get better similarity scores, than those train data users who have rated few dishes in common.
        iv. After all the similarity scores were calculated, I took the top 30 users for each dish my current user had not rated yet and made predictions for my current user based off those top 30 users.

$$\hat{R}_{u'}(o) = \bar{R}_{u'} + \frac{\sum_{u} w_{u,u'}\,(R_u(o) - \bar{R}_u)}{\sum_{u} \left| w_{u,u'} \right|}$$

            1.
            2. For each dish my current user has not rated yet, I used this formula to make a prediction score I think my current user would give that dish
        v. Finally I would keep track of every prediction I made and for each prediction that was a rating of 3, and I would recommend that dish to my current user
2. Average MAE for all test users: 0.5313
3. Precision and Recall for all test users
    a. Average Precision@10: 0.0406
    b. Average Precision@20: 0.0411
    c. Average Recall@10: 0.0192

    d. Average Recall@20: 0.0387

4. I think I am satisfied with my results, when I factored in my ratio when calculating each similarity score It greatly increased my MAE score. My code however was taking a bit too long to run. To get around this problem I used multi processing and threads to speed it up. For each current user or “test_user” I created a new process so the calculations for each user could be done simultaneously.

Part 2:

1. For my custom built model, I built a model that used vector space similarity but between clustered users and dishes
    a. First I used k-means clustering (k = 5) to cluster all my users together into different clusters. I made the number of clusters variable depending on the size of the corpus
        i. Each cluster was based off of users who gave similar ratings in comparison to each other. (Pearson Correlation Coefficient Similarity)
    b. Next, I used k-means clustering (k = 2) to cluster my dishes to together, also with variable clusters frequencies
        i. Each cluster was based off of the presence or absence of an ingredient
    c. After both dishes and users were clustered.
        i. for each current user I would find which cluster of users my current user most similar to. And for all users in that cluster I would take the average score for each dish that my current user had not rated yet. I would save that rating
        ii. For each dish that my current user had not rated yet. I would find which cluster that dish was most similar to. Then I would take the average rating across all dishes in that cluster. I would save that rating.
        iii. For each prediction I would take the average between those two ratings to predict what my user would give that dish
2. Average MAE for all test users: 0.5911
3. Precision and Recall for all test users
    a. Average Precision@10: 0.0446
    b. Average Precision@20: 0.04109
    c. Average Recall@10: 0.0208
    d. Average Recall@20: 0.0384
4. Though this model seems to have performed better than my last model, it definitely is not as consistent. Since my clustering relies on choosing a random position to start each cluster at, the accuracy of my clusters changes everytime I run my program. Sometimes when I run it I will get better scores, and sometimes I will get worse. I think I could have combined some elements from my last model to further improve this one.

Part 3:

1.

	memory-based-model	custom-clustering
Average Mae	0.5313	0.5911

Precision @ 10	0.0406	0.0446
Precision @ 20	0.0411	0.04109
Recall @ 10	0.0192	0.0208
Recall @ 20	0.0387	0.0384
Speed (for all test users)	8.3 seconds	9.77 seconds

2. I think that both models are relatively similar when it comes to performance. I got a little better results on this try on my custom clustering model. However When I scale my corpus of users and recipes up. I more consistently get better results on my memory based model. I get better speeds every time on my memory based model. I think the precision and recall are both low because the average rating for most of the test users was above or around a 3 and most recipes returned did not have a significant difference between them
3. My memory based model works better when there are a lot of users who have rated almost every dish in the corpus. This would mean my similarity scores would be more accurate and therefore my predictions would also be more accurate
4. My clustering model would perform better if most users were almost exactly the same when in comparison to other users in their cluster, and the same goes for dishes and their clusters. My clusters would be less random and would result in more accurate predictions
5. My memory based model cannot make a prediction for a new dish that is added to the dataset. I could alter my clustering model to be only concerned with clustering dishes together and not users. Then my clustering model would be able to make a prediction for this dish.