CSE258 Assignment 1

Henrik Larsson Hestnes
Kaggle display name: Henrik Larsson Hestnes
Kaggle username: henriklarssonhestnes

November 2021

1 Cook prediction

For this task I tried a lot of different approaches, as I was not quite sure what would be the best approach. My baseline solution was a simple "predict true if recipe is popular enough", which actually worked quite well. I tried combining this with Jaccard similarity, both between recipes and users, without any great success.

After that I started implementing a One-Class model, more specifically a Bayesian Personalized Ranking model using Tensorflow. After trying out a wide range of parameters with regards to learning rate, K, lambda and iterations, I found a model which performed reasonably good on my validation set. I then started improving the model by taking cold start problems into account, and predicting these through a popularity-based and Jaccard-based measure, like the extended baseline implementation. After getting an accuracy I was relatively satisfied with on the validation set, I re-trained the model on the whole dataset, before predicting the test set.

1.1 Kaggle performance

	Public leaderboard	**Private leaderboard**
Accuracy	70.3 %	70.4 %
Ranking	566/1121	586/1121

2 Rating prediction

For this task I also tried out a lot of different approaches. I started out using the "Surprise" library to implement a Latent Factor Model. I did not quite get the MSE I was hoping for using this library, so I went on to try implement another Latent Factor Model using Tensorflow. I unfortunately did not manage to get the MSE I was hoping for with this approach either.

I then went on to a Latent Factor Model with only the bias terms on the form

$$rating(user, item) \simeq \alpha + \beta_{user} + \beta_{item} \tag{1}$$

where the optimization problem then becomes

$$\arg\min_{\alpha,\beta} \sum_{u,i} (\alpha + \beta_u + \beta_i - R_{u,i})^2 + \lambda \left[\sum_u \beta_u^2 + \sum_i \beta_i^2\right] \tag{2}$$

By solving the following equations iteratively until they converge, I got the basis for my predictor

$$\alpha = \frac{\sum_{u,i \in train} (R_{u,i} - (\beta_u + \beta_i))}{N_{train}} \beta_u = \frac{\sum_{i \in I_u} R_{u,i} - (\alpha + \beta_i)}{\lambda + |I_u|} \beta_i = \frac{\sum_{u \in U_i} R_{u,i} - (\alpha + \beta_u)}{\lambda + |U_i|} \tag{3}$$

I then used some time to tune my predictor, starting with a very coarse-grained search with different lambdas, before narrowing the search down to find a good lambda on the validation set. I then did some small adjustments, like making sure the predicted rating was saturated from above at 5 and from below at 0, since exceeding this range only can lead to a greater MSE. Then the model was re-trained on the whole dataset, before predicting the test set. I then uploaded my predictions to Kaggle, with a result I am very satisfed with.

2.1 Kaggle performance

	Public leaderboard	**Private leaderboard**
MSE	0.82151	0.79289
Ranking	88/552	33/552