Final Exam - Part 2

12 questions

1.

Why would someone use RMSE (Root Mean Squared Error) instead of MSE (Mean Squared Error) or MAE (Mean Absolute Error)?

- RMSE can be negative or positive, while both MSE and MAE are always positive.
- RMSE is expressed in the same units as the ratings, unlike MSE.
- RMSE is expressed in the same units as the ratings, unlike MAE.
- RMSE penalizes all errors the same, regardless of size, while MAE penalizes large errors more than small ones.

2.

When computing serendipity, we depend upon a prior "primitive" estimate of obviousness and a determination of whether a recommended item is actually relevant. Why do we need these measures?

- Because serendipity is measuring the degree to which an algorithm is giving recommendations for non-obvious, but still relevant, products or items.
- Because serendipity scores are measuring how broad a set of items can be recommended -- for instance, recommending books from different

genres and authors.

- Because serendipity is trying to measure the degree to which an algorithm recommends things the user doesn't want, but that the system is trying to push or sell to the user.

- Because serendipity scores measure the degree to which a user has tastes that differ from the average overall user taste--i.e., to which the user prefers non-popular items.

3.

What is the **major** problem of offline evaluation with unary data?

- Users don't really express unary preferences; just because somebody bought two things doesn't mean she liked them equally well.

- It is usually too hard to obtain unary data because users don't understand the concept.

- If the recommender picks something the user didn't purchase, we do not know if they didn't like it (bad recommendation) or didn't know about it (potentially great recommendation).

- It is too hard to compute metrics with unary data.

4.

The item-item model is often truncated to only keep *M* neighbors per item. The scorer uses at most *k* neighbors to compute each prediction. Why must *M* be significantly larger than *k*?

- Because the M neighbors we keep may not be good ones, so we need to find the top-k best neighbors from among those M.

- [x] Because the user we're predicting for may not have rated all M of the neighbor items; we need enough potential neighbors to be able to find *k* of them among the user's ratings.
- [ ] Because item-item collaborative filtering depends on reduction of the matrix rank to rank-k as a mechanism for smoothing out artifacts in the data set.
- [ ] Because some of the M neighbors we pick will already have been rated by the target user, and we need to find k unrated ones to recommend to that user.

5.

Massa and Avesani's trust-aware recommender multiplies user vectors by their trust weight prior to doing item-item collaborative filtering. What does this accomplish?

- [ ] It adjusts for the fact that the user vectors were originally divided by trust weight to restore all ratings to equal weight.
- [ ] It decreases the undue influence of users who rate many items.
- [x] It makes high-trust users' ratings more influential in computing item similarities.
- [ ] It removes low-trust users from the system.

6.

What is the goal of probabilistic latent semantic analysis (PLSA)?

- [ ] To determine the best set of keywords or product attributes to use in a recommender system.

- To compute a non-personalized probability distribution of preferences across all keywords or product attributes.
- To compute $P(i|u)$ (the probability that user u will purchase/click/read/like i) by decomposing that probability into latent factors.
- To compute $(P(u|i)$, the probability, given that an item i was purchased/clicked/read/liked, that the person who did that was the user u by decomposing both users and items into latent factors.

7.

When holding out ratings from a user's profile for evaluation, what is the benefit of holding out the last ratings rather than holding out random ratings?

- It prevents us from evaluating performance on the user's earliest ratings, which usually aren't very meaningful anyway.
- It more accurately simulates the recommender's knowledge when the held-out ratings were given.
- The most recent ratings have the least information in them, so we don't lose as much accuracy as we would if we held out earlier ratings.
- It makes the evaluation more deterministic. Non-deterministic evaluation is inherently less useful.

8.

What is folding-in in a singular value decomposition?

- Further reducing the number of dimensions for an already factored and reduced matrix.
- Adding additional dimensions into an already factored and reduced matrix to compensate for errors found in recommendations.
- Incorporating new ratings, users, or items into the model by using the existing factorization and updating/computing a feature vector from the ratings.
- Adding new item attribute data into the model by translating item attributes into vector-space features.

9.

In module 5 we taught about many techniques for evaluation. We also pointed out that most evaluation techniques do not address the question of whether the items recommended are actually useful recommendations. Instead, those evaluations focus on whether the recommender is successful at retrieving “covered up” old ratings. Which of the following evaluation metrics successfully focuses on whether the recommender can produce recommendations for new items that haven’t already been experienced by the user?

- Accuracy metrics such as RMSE
- Decision-support metrics such as top-N precision
- Rank metrics such as nCDG
- None of the above

10.

Which of these best explains how we obtain personalized predictions for a target item i, and a target user u using item-item collaborative filtering?

- [x] We identify which of the close item-neighbors to i have been rated by u, and we compute a weighted average of those ratings, weighted by how similar the items are to i.
- [ ] We identify a set of other users who've rated item i, and use item-overlap to determine which of those are closes to target user u. Then we average their ratings for item i, usually using normalized ratings.
- [ ] We average all of the ratings given to item i, but using a weighted average where the weight of each rating is based on the similarity between the user who gave that rating and the target user u.
- [ ] We compute the normalized rating for item i by subtracting from each rating the mean rating given by that user. Then we personalize that normalized rating into a prediction for user u by adding it to u's mean rating.

11.

There's no reason SVD-based approaches to recommendation are guaranteed to reduce dimensionality, but in the end they almost always do substantially. Mathematically, that's because:

- [ ] Factoring a matrix almost always results in a lower-rank approximation of the original matrix.
- [x] Most of the singular values are almost always very close to zero, indicating that not much is lost by reducing those dimensions.
- [ ] The dimensions produced by SVD are guaranteed to be orthogonal.

- After factoring the matrix, there are separate representations of users and of items in different matrices.

12.

Which of the following would not be considered a context in context-aware recommendation?

- A user's emotional state or current mood.
- A user's current location.
- A user's product preferences
- What time or date it is.

Submit Quiz

(https://accounts.coursera.org/i/zendesk/courserahelp?return_to=https://learner.coursera.help/hc)