Final Exam - Part 1

12 questions

1.

Why would you use a different metric for evaluating prediction vs. top-N recommendation?

- Because you need different algorithms to compute predictions vs. top-N recommendation.
- Because predictions are a harder problem; recommendations are just suggestions and can never be wrong.
- Because predictions are mostly about accuracy and error within a particular item, while top-N is mostly about ranking and comparisons between items.
- Because prediction metrics are usually on a 1 to 5 scale, and you need a larger scale for top-N metrics.

2.

Which of the following is an advantage of nDCG *compared with Spearman rank correlation*?

- In nDCG Large moves (e.g., off by 10 positions) are penalized more than small moves.
- Ranking accuracy at the top of the list is weighted more heavily than accuracy further down the list.

- nDCG only considers the order of items, not the numeric scores the recommender gives them.
- nDCG doesn't care about the range of ratings a user uses.

3.

Which of the following statements about diversity metrics is **not true**?

- Diversity measures how much the top items in a recommendation list vary.
- Diversity metrics generally use a separate measure of similarity, either pairwise or for a list.
- A recommendation list with high diversity will have a mix of highly-scored and lower-scored items near the top.
- The goal of measuring and tuning diversity is to prevent over-specialization into a narrow portion of the product or item space.

4.

In some top-n evaluations, instead of considering all items, the recommender recommends from the items the user has rated/consumed/purchased plus a random subset of all items. Why is this useful?

- The evaluation can only judge whether the returned items are among the rated/consumed/purchased ones; having too many other items just increases the number of desirable but not-yet-consumed items, making it harder to tell whether the recommendations are good.

- Most recommender algorithms are more accurate over a smaller set of candidate items, so this reduction makes it easiest to obtain a desired level of accuracy.
- It is useful to evaluate both accuracy and decision-support in a recommender. If all of the items are available to recommend, that makes the user's decision-making much harder.
- To make the results a well-formed random sample for statistical analysis purposes.

5.

Why is item-item more amenable to pre-computation than user-user?

- When there are many more users than items, item similarities are fairly stable, while user similarities can change rapidly as the user interacts with the system.
- When there are many fewer items than users that means there are many fewer correlations to pre-compute.
- Because item-item tends to exhibit lower serendipity, and therefore less popular items don't matter much.
- Because items don't really have correlations; all the correlations are made through the users.

6.

What is a hybrid recommender?

- A recommender that uses both unary ratings and ratings on a multi-point scale.

- A recommender that combines 2 or more algorithms into a single recommender.
- A recommender that produces both recommendations and rating predictions.
- A recommender that recommends multiple types of items.

7.

Most values of the user-item rating matrix are unknown, but traditional SVD-solving algorithms need a complete matrix. What is **not** a good way to satisfy this requirement in a situation where the user ratings are on a 1 to 5 scale?

- Use the user's mean rating.
- Use the item's mean rating.
- Leave it 0.
- Subtract user, item, or personalized mean from the rest of the rating matrix and use 0 for the unknown residuals.

8.

In addition to providing good recommendations, matrix factorization techniques can improve privacy and robustness over nearest-neighbor techniques. Why?

- Individual user preferences are not represented in the factored matrices.
- The relationship between input and output is less direct, going through latent features, and is harder to directly manipulate.

- User preferences in matrix factorization approaches are always normalized, so it is easier to deny liking an item.
- The matrices representing user preferences are stored on a secure server, so manipulation or hacking is much less likely.

9.

What is the key idea of learning-to-rank?

- Show the user various items for the purpose of measuring their response and updating the recommender's model accordingly.
- Directly optimize the recommender to pick good items rather than just score items accurately.
- Generate recommendations by randomly perturbing the items with the highest predicted ratings.
- Deliberately introduce diversity into the list of recommended items to improve user satisfaction and retention.

10.

In the mid 1990s, Net Perceptions was struggling to meet performance goals with its user-user collaborative filtering algorithm. What was its solution to this problem?

- Limit the number of users handled to no more than 250,000 at a time.
- Mining through user profiles to find clusters of similar users and combining their records.

- [x] Switching from user-user collaborative filtering to an item-item algorithm.
- [ ] Switching from user-user collaborative filtering to a matrix-factorization algorithm.

11.

When would it not make sense to use item-item collaborative filtering (compared with user-user)?

- [ ] When different users have very different tastes.
- [x] When there are many more items than users.
- [ ] When there are lots of similar items.
- [ ] When you're more concerned about prediction than about recommendation.

12.

Suppose we wanted to use SVD-based approaches to represent person attributes and item affinities for people, rather than the other way around. In other words, we want each item vector to represent an item's suitability for different person-types, and each person vector to represent the degree to which that person exhibits those person types. Which of the following changes would we need to make to accomplish this goal.

- [ ] We would need to build a set of person-attributes to attach to each user, and then include those in the SVD algorithm.
- [ ] We would need to invert our matrix before running it through SVD.
- [ ] We would need to factor the matrix differently, so we ended up with items in the left-hand-factor.

We wouldn’t need to change anything at all; SVD already does this.

Submit Quiz

Help Center

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