Recommender Systems Specialization Capstone

Part I: Designing a Measurement and Evaluation Plan

- *Translation of business goals and constraints into metrics and measurable criteria*
  The final objective of the recommender is to increase the sale of office products on Nile-River.com during the Back-to-school period. We have also been given specific requirements or constraints for the recommenders which will drive our choice of metrics to evaluate the recommenders. We have also been told that there are two places on the landing page for recommendations. We will consider each of them separately as follows:
    - Since the goal is to recommend items rather than predict preferences, we should focus heavily on decision support and ranking metrics.
    - As far as possible, we want to make sure that the items we recommend are relevant to the user. Hence, **Precision** and/or **Recall @ N** might be god metrics to make sure that the items that come up at the top are the most relevant
    - Research has shown that additional sales during the back-to-school period are divided broadly between traditional school products as well as office products. It has also been observed that large purchases are a combination of inexpensive and expensive products. **Diversity** of the product type and diversity in price seem to be a good metric to ensure we cover all types of products in the recommendations.
    - Nile-River.com also prides itself in having a large product catalog and wants to introduce users to items that they wouldn't find in traditional stores. Hence, the **Serendipity** metric might also be a good one to consider.
    - On a similar note, if there is an interest in reaching the entire product catalog, **Coverage** might also be a good background metric to decide what items to not to recommend because of popularity.

- *A plan for evaluating a set of base algorithms*
  Since this is NOT an Honors track submission, we will not be talking about how to implement and tune the algorithms. Instead, I will elaborate on which algorithms I am choosing based on the ones provided and why.

    - Content Based Filtering: CBF algorithms are the simplest form of personalized algorithms and might be good as a baseline to compare against more sophisticated algorithms involving collaborative filtering and matrix factorization.
    - Collaborative Filtering: Given that we have user and item profiles, we should definitely leverage collaborative filtering algorithms in order to recommend products to users based on similar user/item profiles. However, when comparing user-user and item-item collaborative filtering, we know that item-item collaborative filtering scales better and doesn't face the cold-start problem of adding new users as in user-user collaborative filtering. Hence, we will choose Item-item collaborative filtering for evaluation.
    - Matrix Factorization: Matrix factorization algorithms are more sophisticated than content based and collaborative filtering algorithms since they take into account

latent features addition to the ratings matrix. Hence, this would be a good addition to the recommendation portfolio.

For evaluation, we will go with the hidden data method, where we hold out some data, train on the rest and evaluate on the hold out dataset. K-fold cross validation might not be useful in this case since the data is temporal and we don't want future and past data to intermix while creating splits for cross validation. We also know that sampling the data randomly for training and testing gives worse results. Hence we will do a straightforward 80-20 split of train-test data where the first 80% of the data is used for training and the rest 20% is used for testing.

The steps for evaluating the recommenders will be as follows:

- In addition to the decision support and rank aware metrics, we will calculate one accuracy metric i.e. RMSE.
- To evaluate whether or not items at the top are relevant or not, we will calculate Precision @ N since we want top N items to be relevant. We will test with N=5.
- Finally, we will evaluate the metrics corresponding to the business requirements as specified before. To measure Diversity in price, we will calculate the standard deviation of the products recommended. The higher the standard deviation, the higher the diversity. To measure Diversity in the product types, we will calculate the number of different types of products present in the recommended list. To measure Serendipity, we will calculate the number of items recommended that have less than a certain threshold of ratings, indicating that they are less popular.
- We will average all the metrics across all the users in the test set (since we aren't creating a train-test split here for the non-Honors track, this will be all users), to get overall performances of the recommenders.

- *A plan for constructing and evaluating hybrid algorithms*
  - The problem statement mentions that users reach the landing page in two ways- one, by clicking banner ads for back to school shopping, or selecting the office products category from other navigation aids. Given this behavior, we can have two separate recommenders for traditional school products and office products and depending on how the user reached the landing page, mix the recommendations from the two recommenders i.e. if the user came through the back-to-school route, that might indicate that they more inclined towards purchasing school supplies, so we weight the school supply recommender higher and show more school supplies and lesser number of office products.
  - We can also switch between recommenders, for example, we can default to a collaborative filtering or matrix factorization recommender for "fuller" user and item profiles, but we can switch to recommendations from a content based recommender if an item does not have enough ratings or if we encounter a new user. The final recommendation can be a weighted sum of recommendations from the recommenders, with perhaps a lower weightage from the content based recommender.

Part II: Measurement

The metrics reported in this section have been calculated in this notebook.
The following is a comparison of metrics across the five algorithms provided in the dataset:

Algorithm	RMSE	Precision @ N	Product Diversity	Cost Diversity	Serendipity
CBF	0.572	0.061	9.16	21.001	1.0
UUCF	0.545	0.068	9.09	26.862	1.0
IICF	0.574	0.073	9.29	28.369	1.0
Pers-Bias	0.666	0.075	10.0	5.413	1.0
MF	0.659	0.082	9.23	38.536	1.0

The above metrics were calculated for top 10 items predicted by each of the algorithms across all users in the provided dataset. We chose top 10 because we have a total of 10 spots to recommend items on the landing page.

- Conclusions:
    - In terms of accuracy, UUCF seems to have performed the best with lowest RMSE.
    - Matrix Factorization and Item-item collaborative filtering have the highest precision @ N, which means the items recommended by these systems were most relevant across the user pool.
    - The product diversity of all the algorithms was pretty high, except UUCF was relatively lower
    - Matrix Factorization showed the highest cost diversity and beat the other algorithms by a large margin. It possibly also recommended mostly costly items. It is also noteworthy that the Pers-Bias algorithm predicted mostly cheap items.
    - All the items performed well on the Serendipity front, meaning that they all are pretty good at recommending un-popular/new/unexpected items.

- Next steps:
  Looking at the results above, we will select Item-Item CF, Matrix Factorization and Pers-Bias for hybridization in Part III. While the RMSE of the three algorithms is slightly higher compared to the other algorithms, we will focus on the decision support and rank aware metrics, since our objective is focused more on recommending relevant items than making accurate predictions. We choose Pers-Bias in particular since we've seen that it has recommended mostly cheap items (looking at the Cost diversity metric), which can be used to offset the "costliness" of the IICF and Matrix Factorization algorithms in the hybridization stage.

Part III: Mixing the algorithms

The code for this section is in this notebook under the section named “Hybridization”. We will explore the following hybridization techniques in this section:

- Explore and evaluate:
    - Linear combination of the predictions from the different recommenders- With this kind of combination, we want to see if we can reduce the overall RMSE of our predictions
    - Non-linear ensemble (tree-based) of the predictions from the different recommenders- With this combination as well, we want to improve the accuracy of our predictions
    - Mixing the top-N lists from all recommenders, and creating a list with higher weightage for school products (inexpensive products) if a user reaches the landing page via banner ads for school products and vice versa i.e. higher weightage for office products (expensive products) if the user arrives at the landing page via endorsements for office products.
- Explore only:
  Switching to content based recommender for users with lesser number of ratings and using a combination/other recommenders otherwise.

The evaluation for the hybrids and top-5 recommendations for three users for the three evaluated hybrids can be found in the notebook under the headings “Performance” and “Top-5 for 3 users” respectively.

Part IV: Proposal and Reflection

Proposal

The goal of this project was to develop a recommender system to increase sales of office products on Nile-River.com during the back-to-school period. In order to maximize business values, we were also given some constraints to work with. First, there were two opportunities on the landing page for us to recommend items. Also there was a requirement for the recommended items to have a good mix of costly and inexpensive items. Additionally, Nile-River prides itself in having a more diverse and unique portfolio of products compared to traditional big-box stores. Hence, one of the goals of the recommender was also to introduce users to new products that they wouldn’t expect to find in the usual stores.

In order to evaluate the proposed recommenders, we focused both on the accuracy of the recommenders to make sure users would find our recommendations reliable, as well as decision support and rank-aware metrics to measure relevance of the recommendations to the user and to address the other business requirements like cost diversity, serendipity of the recommended products etc.

For accuracy, we relied on a metric called Root Mean Squared error. This metric gives a measure of how far the predicted rating of the recommender was from the actual user ratings. The recommenders we evaluated were all off by 0.5-0.6 from the original user ratings which is considered acceptable.

We focused on a measure called Precision @ N to gauge the relevance of our recommenders. This metric gives the proportion of items within the recommended items that are relevant to the user, based on items he has rated before.

The business requirements also mention a need for having a good diversity in terms of price and the types of products recommended. For this we use the cost and product diversity metrics. Cost diversity measures the price range of items recommended, while product diversity is a measure of how many different categories of items were recommended. An additional ask was to recommend items that users can't find in big-box stores, hence we also used the serendipity metric, which is a measure of how new or unique the recommended items are compared to what the user has rated before.

The ideal recommender should have a good tradeoff between all the evaluation metrics mentioned above. We tried five different recommenders as part of this capstone- a content based recommender, user-user collaborative filtering, item-item collaborative filtering, matrix factorization and personalization bias. All these algorithms have their own strengths and weaknesses, example the matrix factorization and item-item collaborative filtering have a high cost diversity, which means that they are good at recommending office products, whereas the personalization bias recommender has a smaller cost diversity which means that it's good at recommending inexpensive aka school products. Some algorithms are more accurate than others and we might want to leverage this so the recommendations we make are accurate.

We found that the hybrid recommenders do a good job of balancing trade-offs between the different recommenders. The switching recommender for example could be a good option for cases where we don't have enough ratings for users and items. We could further tune the recommendations based on different user personas, which we demonstrated in the weighted recommender based on how the user got to the landing page, which adds an additional layer of personalization to the experience since we are also using a signal about what the user might be looking for. We have demonstrated both of these in the notebook.

Reflection

- *Reflect on the process of translating business requirements to metrics. What was easy or hard about that process? Do you feel you had adequate preparation in the specialization to take on such a task?*

  The difficult process was the fact that this is the first time since the beginning of the specialization where we have been expected to tie all the learnings together. While I felt like I had all the different pieces to come up with a proposal for the recommender system, it took a while to build a coherent story around the decisions made.

  The easy part was the fact that the requirements were defined in such a way that they could be easily translated into one or more of the metrics we had learnt during the course. For example, the recommenders should have a mix of inexpensive and expensive products, thus pointing to a diversity metric.

- *Reflect on the process of evaluating individual algorithms and creating hybrids. How easy or hard was it to identify differences in performance of algorithms on different criteria? How easy or hard was is to bring together elements from different algorithms through hybrids? How confident are you that your final result is a good algorithm for the problem.*

  The process of picking and choosing among the given algorithms was tough, given that

there is no obvious difference between the metrics on the provided dataset. It was challenging to come up with a hybridization that shows a clear improvement in performance over the standalone algorithms. It would be interesting to see how the suggested hybrid recommender(s) work in an A/B test during an online evaluation where users are allowed to give feedback on the recommendations they receive.

- *Reflect on data management, including the process of separating training and test data. How well were you able to maintain that separation and avoid overfitting your dataset? How confident are you in the generalizability of your result to other similar datasets?*

  I wasn't really able to do anything sophisticated w.r.t. splitting training and test data. I think this is more relevant to the Honors track, where students are actually expected to implement a recommender system. Although I did do some of this for the hybrid recommenders, where I held out 20% of the dataset for evaluation of the regression and tree based hybrid recommenders. It would have been nice to have temporal information of the data to design a temporal evaluation framework for the models.

- *Reflect on the tools you used for this capstone (whether spreadsheet, LensKit, or external ones). Do you feel you had the experience and skill with the tools you needed for the capstone? Do you feel the tool was a good match for the problem (and if not, what would you have preferred)? Please identify areas where the tools were particularly helpful or particularly challenging.*

  I used Python for the entire project. Python is what I use on a day to day and it is the coding language that I envision myself using for designing recommenders in the future. I would have liked to do the Honors version of the specialization but given that I am not fluent enough in Java I decided against it. It would be nice if the Honors track provided students the flexibility to use other coding languages too, like Python.

- *Finally, reflect on the capstone experience as a whole. Did it achieve its goal of giving you one project to bring together the diverse set of materials you learned in this specialization? Do you feel more capable of (or confident in your ability for) taking on applications of recommender systems? Other lessons you're willing to share?*

  The Capstone indeed helped me bring all my learnings together. I am definitely more interested in the field after taking the specialization and am confident of being able to apply what I have learnt to problems in the real world.