# 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.