Quentin Deligny
2262655D

Coursework Report
Recommender Systems

Question 1

In this section, I have implemented four baseline rankings which will be required for future performance comparisons. We use a baseline model to make ranking predictions relying on the following data statistics:

- Average rating: compute the average rating of each item from all the ratings given to this item.
- Number of ratings: represents the total number of ratings given to an item.
- Number of 5-star ratings.
- Fraction of 5-star ratings: number of 5-star ratings divided by the total number of ratings for an item.

These predictions will be constant regardless of the user for which they are made. This is not a personalised approach and is therefore expected to perform quite poorly. We will use the *mrr_score* function to assess the performance of our models throughout this coursework. The MRR Scores computed using the baseline model will serve as reference for measuring the improvements made by future models.

The following figures show the results we have obtained using the 4 mentioned metrics:

Figure 1.a

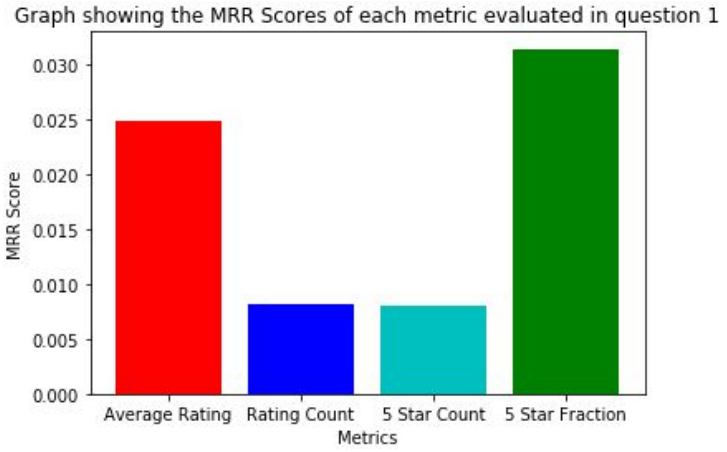
```
1. MRR Score Average Rating:
 0.024775365599441497

2. MRR Score Number Ratings:
 0.008181375050318986

3. MRR Score Number 5 Ratings:
 0.008070258646211246

4. MRR Score Fraction 5 Ratings:
 0.03138372904213526
```

Figure 1.b



From *Figure 1.b*, we can observe that the model relying on the number of ratings or the number of 5 star ratings to make predictions have performed significantly worse than the models using the other two statistics in terms of MRR Score. By looking at *Figure 1.a* for more specific measures, we can see that the predictions relying on the fraction of 5-star ratings has achieved the best performance with an MRR Score of around 0.0314.

We can justify this significant difference in MRR Scores between various metrics by assessing their overall importance and relevance in determining whether or not a specific item is “good” (mostly liked by users) or “bad” (mostly disliked by users). This approach is slightly subjective, but should provide some form of justification regarding the performance of each metric:

- Average rating: a generally good indication on the quality of an item. It is heavily reliant on the number of ratings used to compute this average. The higher the number of ratings, the more accurate the average rating of the item will be.
  Downside: an item with an average rating of 3.5 stars over 100 reviews will be ranked lower than an item with a rating of 4 stars over 2 reviews.
- Number of ratings: a measure of how popular an item is. High popularity does not necessarily imply that an item is "good", but it can be used as an indication (users are generally more likely to read a book which has positive reviews).
  Downside: an item with a high number of bad ratings will be ranked more highly than an item with a lower number of good ratings.
- Number of 5-star ratings: this is somewhat representative of the quality of an item. It is difficult to rank items solely based on their number of 5-star ratings as it is heavily reliant on popularity as well as the total number of ratings.
  Downside: An item with 3 5-star ratings and 100 1-star ratings will be ranked more highly than an item with 2 5-star ratings and 100 4-star ratings.
- Fraction of 5-star ratings: is a generally good assessment of the quality of an item. However, much like the average rating, its accuracy is heavily affected by the number of ratings.
  Downside: same as above.

To summarise, each metric cannot be used by itself to accurately predict the overall ranking of an item as they do not contain enough information. A model relying on a combination of these metrics would certainly achieve better results.

Question 2

In this question, we will define, train and evaluate an explicit and an implicit factorisation model using explicit rating data. We define the models with the same initial parameters to get accurate measures of their MRR Scores. We define the independent variables to be the number of iterations (set to 5), the random seed given to each model (set to 42), the explicit data used as training set, the validation data (used to tune the number of latent factors) and the testing environment (Python Notebooks). The dependent variable is the number of latent factors used by each model to represent each item; this number will vary between 8, 16, 32 and 64 for both models.

We show our results in the following figures:

Figure 2.a

```
MRR Score Explicit for 8 factors: 0.02405349500146899
MRR Score Implicit for 8 factors: 0.1891693862804418

MRR Score Explicit for 16 factors: 0.022389170775864123
MRR Score Implicit for 16 factors: 0.19914659710807783

MRR Score Explicit for 32 factors: 0.03295219038090474
MRR Score Implicit for 32 factors: 0.19726073354137386

MRR Score Explicit for 64 factors: 0.029635651158912788
MRR Score Implicit for 64 factors: 0.16964355193469752
```

Figure 2.b

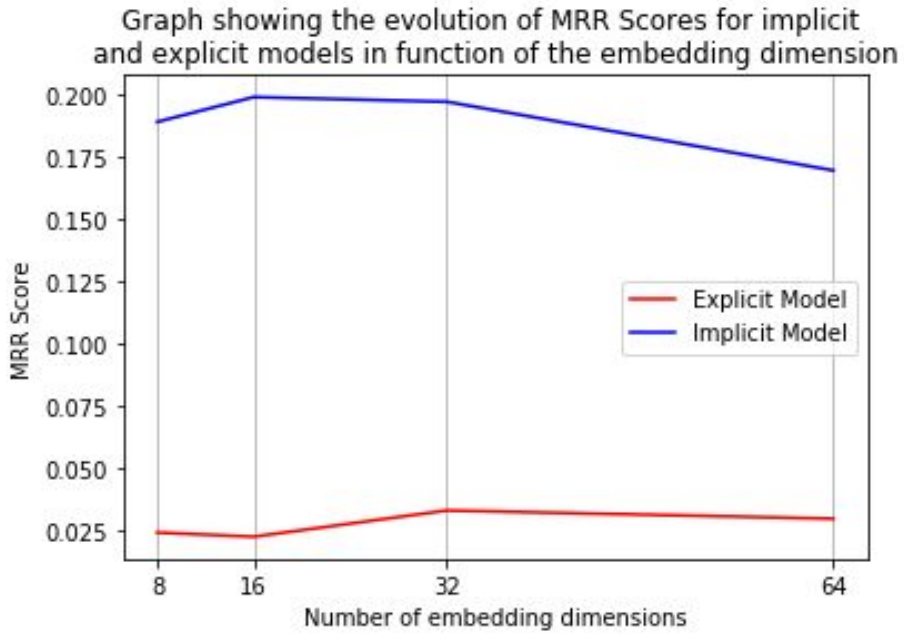


Figure 2.c

```
MRR Score Implicit for 16 factors: 0.3276965863884323
```

As shown in *Figure 2.b*, the implicit model has performed much better than the explicit model in terms of MRR Scores, regardless of the number of latent factors. Our best result (a score of around 0.199) has been achieved using the implicit model with 16 embedding dimensions. On the other hand, the best performance for the explicit model is approximately 0.033. The latter has been achieved given 32 embedding dimensions. After the tuning phase, we use the test data set to evaluate the performance of our best parameter combination. Using the validation set for tuning and the test set for testing is a method used to prevent overfitting of the data by the models. Our final results for question 2 are shown in *Figure 2.c*.

Question 3

The third part is quite similar to our evaluation in the second part, aside from the fact that we will be using implicit data rather than explicit data to produce our predictions. We implement an implicit factorisation model using the same variables as the previous section. Once again, we will examine different performance measurements for different numbers of embedding dimensions.

Figure 3.a

```
LF is  8  - MRR Score:
 0.1931247369716604
LF is  16  - MRR Score:
 0.2031210049469179
LF is  32  - MRR Score:
 0.1910659694926828
LF is  64  - MRR Score:
 0.17840289192215153
```

Figure 3.b

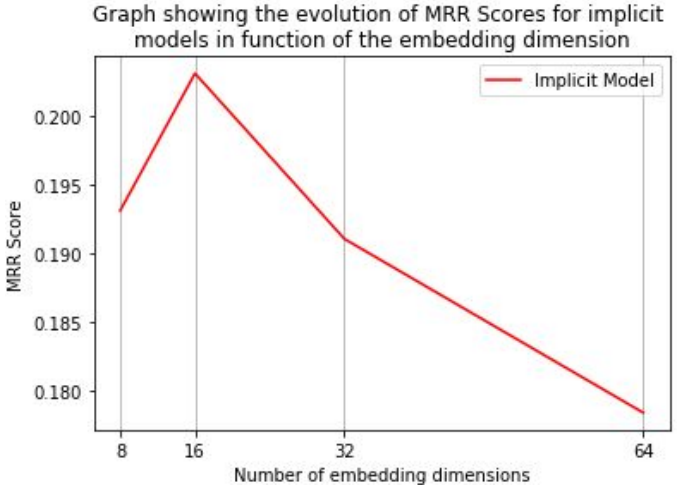


Figure 3.c

```
Best Result:  0.3376630775705313
```

Much like in question 2, we attain our best MRR Scores using 16 latent factors for the implicit model. Despite the change from explicit to implicit training data, we can observe the same evolutions in the line modelling MRR Scores: an upwards trend from 8 latent factors to 16 latent factors (where both curves reach their maximum) followed by a downwards trend from 16 to 64 latent factors. Additionally, both curves have very similar MRR Scores at each point, although the overall scores are slightly higher for the implicit model trained on implicit data (observations made from *Figure 2.b* and *Figure 3.b*). Once again, we proceed by testing the optimal parameter combination found for this task using the test set: the resulting MRR Score is shown in *Figure 3.c*.

For the following steps, we have selected the top 200 users with the highest RR scores in order to further examine the items they have rated (explicit data) and the items they have shelved (implicit data). *Figure 3.d* shows our findings for the top 200 users with the highest scores:

Figure 3.d

```
The most read books among the top 200 users with the highest RR Scores are:
['The Great Gatsby'] was read  48  times
['The Catcher in the Rye'] was read  47  times
['The Hunger Games'] was read  45  times
['To Kill a Mockingbird'] was read  44  times
['Harry Potter and the Chamber of Secrets'] was read  44  times

The most shelved books that have not been read among the top 200 users with the highest RR Scores are:
['The Book Thief'] was shelved but not read  6  times
['The Pillars of the Earth'] was shelved but not read  6  times
['The Glass Castle'] was shelved but not read  6  times
['A Tree Grows In Brooklyn '] was shelved but not read  5  times
["A People's History of the United States: 1492 to Present "] was shelved but not read  5  times
```

The items contained in the implicit data set also appeared in the explicit ratings data set: users read the books they have shelved. In order to find the books that had been shelved by a user but not read, we removed all the books which also appeared in the explicit rating data set from the list of books that had been shelved. This may be an indication that the books have been added to the user's shelf more recently (or that the user shelved but did not prioritise this item). Among the top 200 users with the highest RR Scores, 194 users had a score of 1.0. The top 3 items most often rated by these top users are *The Great Gatsby*, *The Catcher in the Rye* and *The Hunger Games*. These are all popular and highly rated items so explicit data provided for these items "help" the system make more accurate predictions for the concerned user. This may explain why the system has attained high RR Scores for these users. On the other hand, the top 3 items that have been shelved but not read by the top 200 users are *The Book Thief, The Pillars of the Earth* and *The Glass Castle*.

Lastly, we will measure the intra-list diversity of all the users and retain the top 10 and bottom 10 users with the highest and lower intra-list diversity scores. For each of the retained users, we will analyse the top 5 predictions made by the system in an attempt to understand their respective score (the exact details are too large to fit in the report but may be found in the code). We can make some clear observations from the results: the top 5 predicted items for the users with the highest ILD scores are generally much more varied than the top 5 predicted items for users with low ILD scores. While predicted movies are rarely similar for top users (both within predictions made for the same user, and predictions made for all 10 users with the highest score), movies recommended to users with low ILD scores are much more similar. For example, it is not uncommon for Harry Potter movies to be recommended several times in the top 5 recommendations of the same user. Furthermore, there was also a high similarity between movies recommended to different users within the bottom 10 (many appearing between 4 and 6 times).

Question 4

For this stage, we implement a new model aiming to combine the outputs from the previous implicit factorisation models using the *CombSum* data fusion method. This method uses rankings provided by both models to produce a new (theoretically improved) ranking.

Figure 4.a

```
The RR Scores of 582 users have improved from the best model in Q2 to the combSum model
The RR Scores of 569 users have improved from the best model in Q3 to the combSum model
```

The results shown in *Figure 4.a* indicate that the *CombSum* model has improved the RR Scores of 582 and 569 users from the best model implemented in question 2 and 3 respectively. However, this output lacks information to enable us to accurately compare these models. We plotted the following:

Figure 4.b

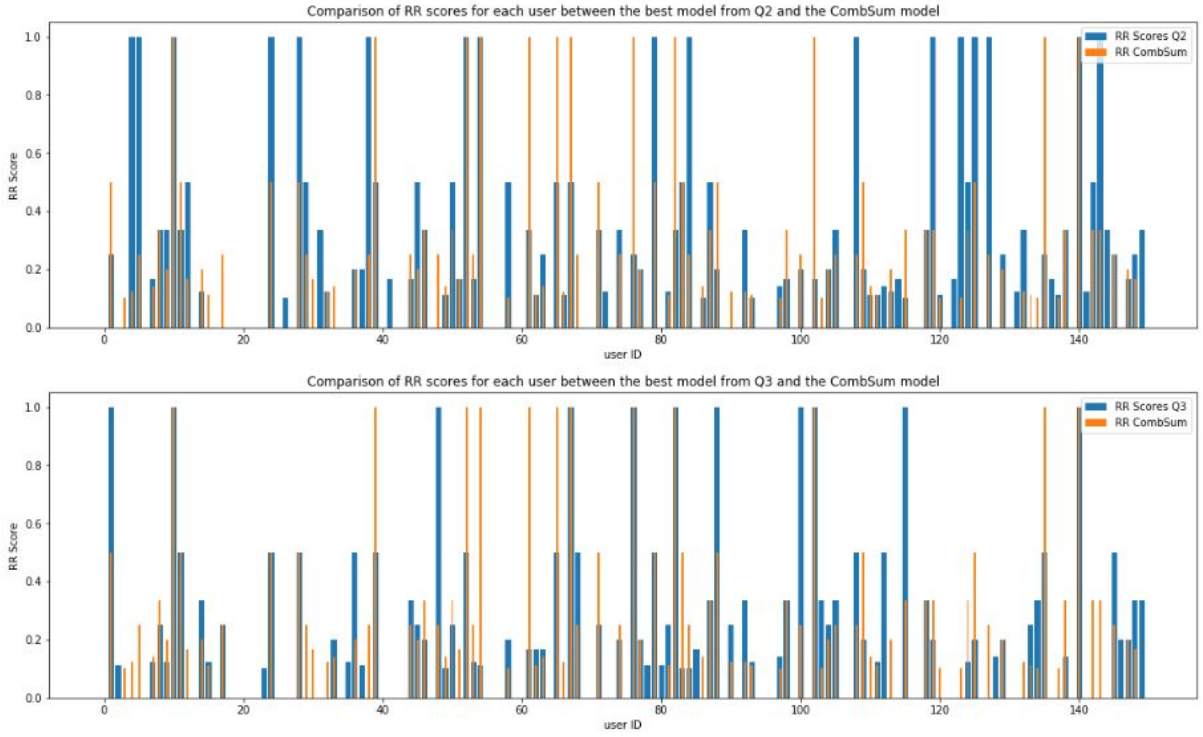
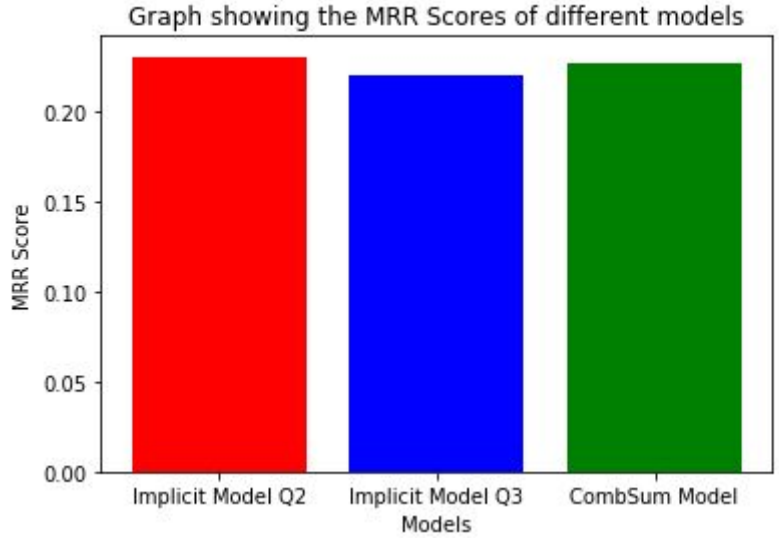


Figure 4.c

```
0.23098791459221674 0.2202132812437965 0.2278224429675155
```



In an attempt to reduce clutter (or size) in *Figure 4.b*, the above graphs only represent 150 users; However we will assume that these 150 users are somewhat representative of the entire population of

users (it does not need to be highly representative as we will only use them to make general observations on the differences in RR scores). The bar charts in *Figure 4.b* highlight the fact that while the *CombSum* model may have improved the RR scores of certain users, many other users had better RR scores in the previous models (from question 2 and 3). This result is understandable, as our current model naively combines the best and worst predictions of each previous model: this approach cannot yield better results than what we already have with our best model.

This figure compares the MRR scores of all the models analysed above, and highlights that there is no real difference (although they each respectively improve the RR scores of various users) between these models. We have not specified any train set when computing the *mrr_score* to avoid getting improper MRR results: since the *CombSum* model relies on both explicit (model Q2) and implicit datasets (model Q3), we could not specify a train set. Specifying a train set affected the resulting scores of each model, so we decided to omit this parameter for the sake of this comparison.

Question 5

This section implements a lift-based recommender system which finds the lift score for all positively rated pairs (pairs rated higher than 4 stars). The lift measures the dependency (or connection) between pairs of items within our dataset. The following table shows our findings:
Figure 5.a

book_id_x	book_id_y	log_lift
1416	418	11.734163
1608	1302	11.734163
1140	1302	11.734163
880	1227	11.734163
1811	1925	11.734163
...	...	...
248	61	1.553044
19	112	1.435261
112	19	1.435261
297	6	1.273119
6	297	1.273119

We can observe that, among the pairs of positively rated books, the log lift varied from 1 (completely unrelated) to around 11.73 which highlights a high correlation. A significant number of pairs within the highest rated books showed a high dependency. The *liftRecommender* model uses these metrics to provide a final score for each item using the sum of the log lift for all positively rated items in the user profile. This model provides a set of predictions based on the given user, but is currently highly inefficient in its current state (could likely be optimised by avoiding the looping over item ids in order to make predictions for a single user given more time).

Question 6

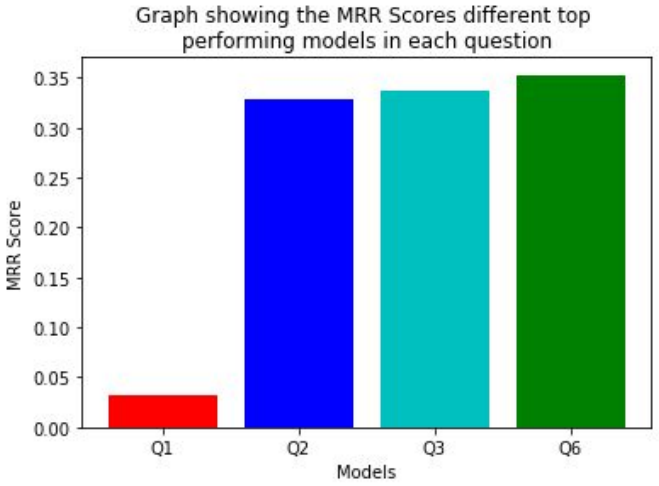
The goal was to design a hybrid model relying on attributing various weights to the previous models. We have selected our best models from the previous questions (ignoring the model from question 5 which is not fully optimised) to combine them with various weights in order to achieve optimal performance. The performance was measured by the MRR score, and the weights were tuned using the validation dataset (implicit). However, the final scores were computed using the test sets.
Figure 6.a

```
The best score achieved by our hybrid model is: 0.3207973034136115.
This score was achieved by combining models from Q1, Q2 and Q3
 The respective combination of weights used is: [0.5, 0.3, 0.2]
```

Figure 6.b

```
Best Model Scores of each question:
Q1: 0.03138372904213526
Q2: 0.3276965863884323
Q3: 0.3376630775705313
Q6: 0.3528107307622065
```

Figure 6.c



Our results indicate that the weights are more or less evenly distributed between all models. Our hybrid model has combined the outputs of the model using the fraction of 5-star ratings to make predictions (our best performing model from question 1), the implicit factorisation model using explicit data from question 2 and the implicit factorisation model relying on implicit data from question 3 by allocating them the following weights respectively: 0.5, 0.3 and 0.2.
Not all weight combinations have been tested (for the sake of time), but the resulting combination was selected for achieving the highest MRR score among 28 other possible weight combinations using the validation set. *Figure 6.b* and *Figure 6.c* summarise our findings throughout this coursework by depicting the MRR scores of our best models for each question.
It may be worth noting that our hybrid model has achieved its best performance by attributing a weight of 0.5 (the highest attributed weight) to the model utilising the fraction of 5-star ratings. This is by far the worst performing model of the three in terms of MRR score. Despite this somewhat surprising selection of weights, our hybrid model manages to slightly outperform all the other models, achieving the best recorded MRR score of 0.353 on the testing data.

Question 7

Lastly, this final section will be used to further discuss and analyse our previous findings in order to establish the different qualities that make a good recommender system for this dataset. The goal is to synthesise our results to reach a general conclusion; We have been using the *mrr_score* function as the only evaluation metric to test our models in this coursework. This implies that we have compared the models in terms of how good they are at ranking items, and may have overlooked other important factors of model evaluation that could provide further information on the overall performance of the model. Therefore, we will assume that a model which achieves a good MRR score can be considered a good recommender system for this dataset (without looking into other evaluation methods).
The first part of this coursework was used to implement various naive baseline models: the model would rely on individual statistics (like number of ratings, average rating, 5-star ratings…) to make general predictions for users. These statistics concerned the items, and the predictions were not tailored to each user. As such, this model was expected to perform rather poorly, and could be used to gauge the improvements brought by future changes. As discussed in the first section, each statistic only provided incomplete data regarding the items in the dataset, which caused it to overlook other important information (which would have improved the accuracy in the prediction of good and bad items). After further discussion of each metric, we have given several reasons as to why the fraction of 5-star ratings and average rating might be our best performing models for this exercise (by far); Not only do they take into account the value of the ratings, they also take into account their total number.

This explains why they may be more efficient than statistics which only consider the number of ratings. However, we could have undoubtedly achieved better results from combining all the different statistics to produce our final item ranking (regardless of users). These assessments help us understand that a good recommender model should make use of all available metrics to produce its final set of predictions.

The following part is used to instantiate and evaluate an explicit factorisation model and an implicit factorisation model. Each is trained on explicit data before being tuned using a validation set. This highlights the importance of tuning our models to ensure they are implemented with the optimal combination of parameters. We tune models using the validation set to avoid overfitting; if we tuned and tested the performance of models on the same set (such as the testing set), they would be more likely to achieve very high scores for this particular set while poorly performing on unseen data. We have tuned only one parameter in this task: the number of embedding dimensions used by each model. Ideally, we would tune all the available parameters used in the model's implementation. However, this is sufficient to show the impact tuning on the final results of our model. A good recommender system for this task should make use of tuning using validation sets and testing to optimise its parameters.

The next section is similar to the previous one aside from the fact that it relies on implicit data to make predictions. As a result, the overall performance (MRR score) slightly increases from the previous best model results. This shows the importance of using the right dataset (depending on the chosen model and the available datasets) for this task. Future analysis will consider how using a combination of different datasets can be beneficial to the performance.

Question 4 and question 6 are closely tied together, so we will discuss our findings for both questions. The key notion introduced is to use the combination of various individual models to produce a new set of results. Question 4 naively combines the outputs of two different models by simply adding their prediction scores. While the notion of a hybrid model is correctly implemented, it is not optimised. The problem is that the predictions of each model is weighted the same, and evaluated only once (no tuning of weights to maximise MRR score). Therefore the new combined models will factor in the good predictions from model x as much as the bad predictions from model y (essentially offsetting any potential improvements). It is impossible for this naive combination approach to improve the model's performance (as the best model's predictions will always be equally "mixed" with the worst model's predictions). Question 6 improves this approach by factoring in weights to the predictions made by each model: this makes our final results more flexible as we are able to evaluate the output of various weight combinations and choose the best results. This step is a good example to illustrate the effectiveness/efficiency trade-off: we can use a greater number of parameter combinations to achieve better results by fine-tuning the parameters at the cost of efficiency. Alternatively, reducing the possible parameter combinations is likely to make our model more efficient but less effective. The general idea is that adding weights enables us to use the best predictions from each model and combine them. This improves the performance of our recommender system for this task.

Finally, although not fully implemented, question 5 helps us understand the importance of other metrics in the assessment of our models. In this specific case, we can conclude that users with higher RR ranks generally have a higher intra-list diversity (less repeated or similar recommendations than among the users with lowest RR score). Intra-list diversity is a useful metric to evaluate how good the model is in this task (we do not want a model which constantly predicts the same items to the users) as it allows for better discovery. Given more time, this question could have been implemented more efficiently: the most notable improvement would be to make use of pandas dataframe operations to compute the sum of the log lift score between positively rated items in the user profile (rating ≥ 4) and the target item rather than iterating over the item ids.