An Investigation into the BPR, EASE and SimSVD algorithms.

Shane Cooke

17400206

Abstract. In this report, I outline the methodologies and details of three different recommendation algorithms, Bayesian Personalised Ranking (BPR), Embarrassingly Shallow Auto-Encoders (EASE) and Singular Value Decomposition (SimSVD). I then attempt to fine-tune and optimise these algorithms, with the goal of comparing the results and assessing which algorithm is best-fit for the movie recommendation problem. Upon documenting these results, I carry out an advanced analysis into the potential for bias and unfairness within the SimSVD recommender system, and discuss many potential remedies to these biases.

1 Introduction

Creating accurate and efficient personalised recommendations is a massive area of research in the online sphere today. In order to create a successful recommender system, precise and effective algorithms must be employed to efficiently identify links and commonalities between data points, which then allows a system to provide user-specific recommendations based on historic data.

For the purpose of this paper and the movie recommendation problem, I have decided to focus on three algorithms, BPR (Bayesian Personalised Ranking), EASE (Embarrassingly Shallow Auto-Encoders), and SimSVD (Singular Value Decomposition). The BPR algorithm falls under the matrix factorisation class of recommender systems, while the EASE and SimSVD algorithms fall under the item-based collaborative filtering class of recommender systems. In the process of completing this report, I aim to outline the intricacies and methodologies of the three algorithms in the process of producing recommendations. I also aim to fine-tune and optimise these algorithms to produce the best possible results for the movie recommendation dataset, and once optimisation has been completed, I aim to compare the sets of evaluation metrics and determine which algorithm is best fit for the movie recommendation problem. Finally, once the optimal algorithm has been determined, I aim to carry out a deep analysis on this algorithm so as to determine the effects of bias and unfairness on the results produced by the recommender.

2 Problem Definition

The algorithm analysis section of this report will focus on a comparison between the performance of the rating prediction algorithm approach and the Item-Based Collaborative Filtering approach when applied to the movie recommendation problem. Both recommender systems will be fine-tuned to produce the most optimised results possible, and both systems will then be evaluated using a number of evaluation metrics such as precision, recall, average relevance of ratings, and the number of common recommendations. I then aim to compare these evaluation metrics, and come to a definitive conclusion as to which system is optimal for the movie recommendation problem.

3 Methodology and Algorithms

For this report, I have decided to focus on and analyse three algorithms which form the backbone of the rating prediction and item-based collaborative filtering approaches. For the rating prediction approach, I have focused on the BPR algorithm, and for the item-based collaborative filtering approach, I have focused on the EASE and SimSVD algorithms, the details of which are discussed below.

3.1 BPR

BPR or Bayesian Personalised Ranking, is a pairwise personalised ranking loss that is derived from the maximum posterior estimator[1]. The training data for the BPR algorithm is constructed using tuples in the form *(u, i, j)*, which represents that the user u prefers the item i over the item j. The Bayesian formulation of BPR which aims to maximise the posterior probability can then be given as

$$p(\Theta| >_u) \propto p(>_u |\Theta)p(\Theta) \tag{1}$$

where Θ denotes the parameters of the recommendation model and $>_u$ denotes the desired personalised ranking of all items for user u. The maximum posterior estimator for the derivation of the generic optimisation criterion for the personalised ranking task can then be given as

$$\begin{aligned} BPR-OPT :&= lnp(\Theta| >_u)lnp(>_u |\Theta)p(\Theta) \\ &= ln \prod_{(u,i,j\in D)} \sigma(\hat{y}_{ui} - \hat{y}_{uj})p(\Theta) \\ &= \Sigma_{(u,i,j\in D)} ln\sigma(\hat{y}_{ui} - \hat{y}_{uj}) + lnp(\Theta) \\ &= \Sigma_{(u,i,j\in D)} ln\sigma(\hat{y}_{ui} - \hat{y}_{uj}) - \lambda_\Theta||\Theta||^2 \end{aligned} \tag{2}$$

where D is the training set denoting the items the user u liked, and $\hat{y}_{ui}$ and $\hat{y}_{uj}$ are the predicted scores of the user u for the items i and j respectively[2].

3.2 Item-Based Collaborative Filtering

Item-based collaborative filtering is a system in which user-rated items are matched to other relevant items by first computing item to item similarity, and then computing a recommendation prediction based on this similarity. For the item to item similarity aspect of this system, I have focused on two main computational methods which are known as the EASE (Embarrassingly Shallow Auto-encoders) and SimSVD (Singular Value Decomposition) similarity learners.

The EASE algorithm produces a solution which is obtained through the inversion of the Gramian matrix $R^T R$. In the case of binary data, the *(i,j)* component of the Gramian is just the co-occurrence count, or the number of users who rated both item i and j. EASE then suggests that the inverse is the right object data structure on which to base the similarity, rather than the Gramian itself. The EASE solution is asymmetric and can be given as

$$S^* = (R^T R + \lambda I)^{-1}(R^T R - D(\lambda)) \tag{3}$$

where D is a diagonal matrix, with the Langrange multipliers λ on the diagonal.

The SimSVD algorithm produces a solution based on the Singular Value Decomposition of an $m\ x\ n$ real matrix R, which can be given in a factorisation of the form

$$R = U \Sigma V^T \tag{4}$$

where U is an $m\ x\ m$ real unitary matrix, Σ is an $m\ x\ n$ diagonal matrix with non-negative real numbers on the diagonal, and V is an $n\ x\ n$ real unitary matrix. Singular value decomposition gets its name from the diagonal entries on Σ, which are called the singular values of matrix R. They are in fact, the square root of the eigenvalues of matrix $R^T R$. Just like a number factorized into primes, the singular value decomposition of a matrix reveals a lot about the structure of that matrix, which can be utilised effectively in an item-based system[3].

The output from the EASE and SimSVD similarity learners can then be used in conjunction with a recommendation score producer, to compute the similarities between items in a dataset, completing the item-based collaborative filtering system.

3.3 Fine-Tuning and Optimisation

Each of the three algorithms discussed above require a number of configuration variables in order to facilitate proper implementation. Some of these configuration variables are unique to a specific algorithm, and some are general configuration variables exhibited in all three algorithms. The N and *nusers* variables are exhibited in each algorithm, and for the purpose of fair comparison, I decided upon an N value of 20 and an *nusers* value of 500 across all algorithms.

The BPR algorithm uses a *dim* variable which specifies dimension sizes, and a *latentSpaceDim* variable which specifies the size of the latent space dimension.

The EASE algorithm uses a *lambda* variable as a regularisation parameter, and the SimSVD algorithm uses *mu* and *K* variables as regularisation parameters. As you can see, this wide array of potential configuration variables leaves a large amount of room for algorithm fine-tuning and optimisation, the results of which are discussed in the Evaluation section below.

4 Evaluation

Below I will provide an outline and evaluation of the baseline results achieved by each algorithm, the optimised results achieved by each algorithm, and a comparison between the results achieved by the BPR, EASE and SimSVD algorithms.

4.1 BPR

The BPR algorithm takes *dim* and *latentSpaceDim* configuration variables which allowed for some amount of optimisation and fine-tuning. The first configuration variable that I analysed was the *dim* variable. As you can see in Figure 1, I began with a *dim* value of 5, and increased this value in steps of 5 up to and including a *dim* value of 60. I then noted the precision and recall results achieved, and plotted them on a line chart for analysis, as can be seen in Figure 1 below.

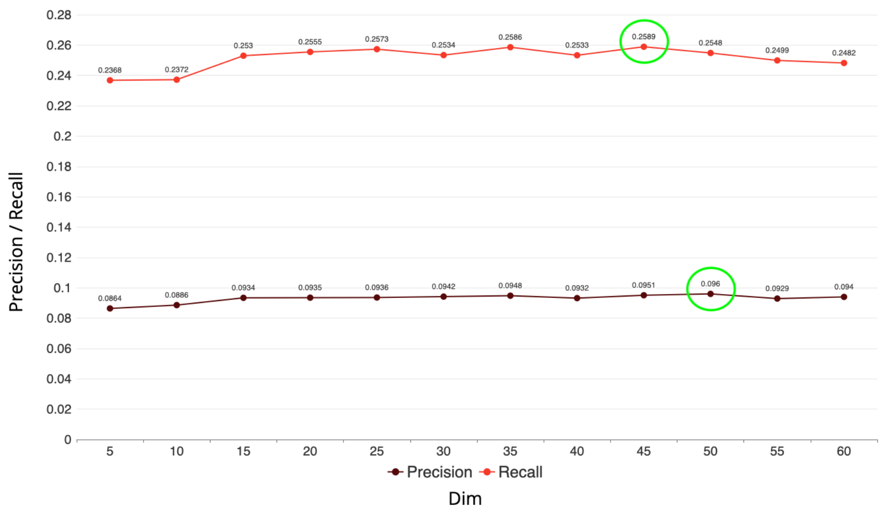

Fig. 1. A graph showing the Precision and Recall achieved by the BPR algorithm, dependent on the *dim* variable value.

A *dim* value of 50 achieved the highest precision (0.096) and a *dim* value of 45 achieved the highest recall (0.2589). These results exhibited an increase of 0.0096 precision, and 0.0221 recall over the unoptimised BPR algorithm, which shows a noticeable degree of successful optimisation through configuration variable tuning. The highest achieving *dim* value overall came to a value of 45 when taking into account both precision and recall, and when the *dim* variable value increased above 50, both evaluation metrics began to decrease steadily. Varying the *latentSpaceDim* configuration variable however, had no positive or negative effect on the precision or recall returned by the algorithm, no matter how drastic a change I instituted.

4.2 Item-Based Collaborative Filtering

EASE

The EASE algorithm takes *lambda* as a configuration variable, which again allowed for some amount of optimisation and fine-tuning. As you can see in Figure 2, I began with a *lambda* value of 0.1, and increased this value in steps of 0.1 up to and including a *lambda* value of 1.00. I then noted the precision and recall results achieved, and plotted them on a line chart for analysis.

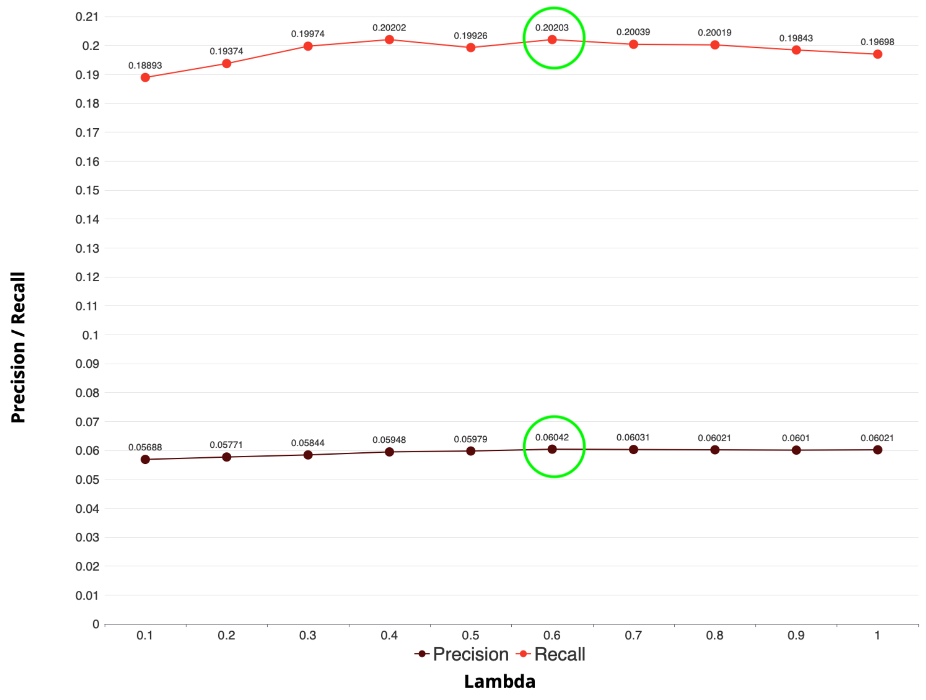

Fig. 2. A graph showing the Precision and Recall achieved by the EASE algorithm, dependent on the *lambda* variable value.

As you can see in Figure 2 above, a *lambda* value of 0.6 achieved the highest precision (0.06042) and a *lambda* value of 0.6 also achieved the highest recall (0.20203). These results exhibited an increase of 0.00354 precision, and 0.0131 recall over the unoptimised EASE algorithm, which again shows a noticeable degree of successful optimisation through configuration variable tuning. When the *lambda* variable value increased above 0.6, both the precision and recall results achieved began to decline steadily, showing that 0.6 is in fact the optimal value for the *lambda* regularisation parameter.

SimSVD

The SimSVD algorithm takes K and mu configuration variables which also allowed for some amount of optimisation and fine-tuning. The first configuration variable that I analysed was the K variable. As you can see in Figure 3, I began with a K value of 5, and increased this value in steps of 1 up to and including a K value of 25. I then noted the precision and recall results achieved, and plotted them on a line chart for analysis.

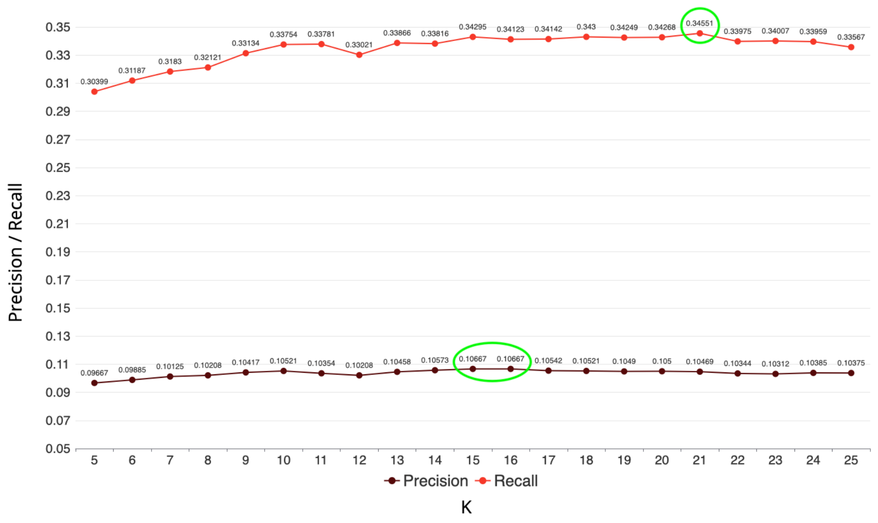

Fig. 3. A graph showing the Precision and Recall achieved by the SimSVD algorithm, dependent on the K variable value.

A K value of 15 and 16 achieved the highest precision (0.010667) and a K value of 21 achieved the highest recall (0.34551). These results exhibited an increase of 0.01 precision, and 0.04152 recall over the unoptimised SimSVD algorithm, which like the previous two algorithms, also shows a noticeable degree of successful optimisation through configuration variable tuning. Precision began to decrease steadily when the value of K increased above 15, and recall began to steadily decline when the value of K increased above 21. Taking both precision and recall into account, a K value of 15 is optimal for the SimSVD algorithm. Varying the mu configuration variable however, had no positive or negative effect on the precision or recall returned by the algorithm, no matter how much the variable was increase or decreased in size.

4.3 Comparison

In order to compare these three algorithms I decided to focus on three main research questions, "Which algorithm responds best to fine-tuning and optimisation?", "Post-optimisation, which algorithms exhibited the most common recommendations?", and "Post-optimisation, which algorithm exhibits the best results overall?".

RQ1: Which algorithm responds best to fine-tuning and optimisation?
Each of the three algorithms that I have analysed in this report exhibited some amount of success in the fine-tuning and optimisation process, however some exhibited more success than others. Figure 4 below shows the % change in both precision and recall for each of the three algorithms post fine-tuning and optimisation. As you can see, the BPR algorithm exhibited the highest % change in precision, with a change of 11.11%. This change was significantly higher than the changes exhibited in EASE (6.22%) and slightly higher than the changes exhibited in SimSVD (10.34%). For recall, the SimSVD algorithm exhibited the highest % change with a 13.66% improvement. This was significantly higher than the change observed in EASE (6.93%), and also significantly higher than the change observed in BPR (9.33%).

The SimSVD algorithm exhibited the best overall response to fine-tuning and optimisation with a combined % change of 24%. The BPR algorithm followed in close second with a combined % change of 20.44%, while the EASE algorithm exhibited comparatively weak optimisation results with an combined % change of only 13.15%.

RQ2: Post-optimisation, which algorithms exhibit the most common recommendations?
Due to the fact that the EASE and SimSVD algorithms fall under the same class of recommender system (item-based collaborative filtering), I had assumed that these two algorithms would exhibit the highest amount of common recommendations. This however was not the case, as BPR and SimSVD exhibited the highest amount of common recommendations by a large margin, with a total of 11.155 similar recommendations out of 20 possible recommendations. The second highest was BPR and EASE with 9.793 common recommendations out of a possible 20, and EASE and SimSVD exhibited the lowest amount of common recommendations, with only 8.449 out of a possible 20.

BPR and SimSVD exhibited a 55.78% common recommendation rate, while EASE and SimSVD only exhibited 42.25% common recommendation rate. This result was surprising, as I had previously assumed that the EASE and SimSVD algorithms would return a much higher common recommendation rate due to the fact that they are from the same class of recommender system (item-based collaborative filtering). The results however disproved this assumption, as BPR and SimSVD, two algorithms with widely differing methodologies, exhibited the highest common recommendation rate overall.

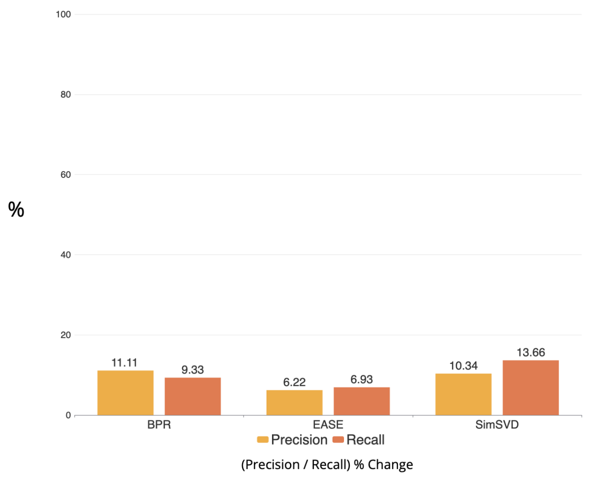

Fig. 4. A graph showing the % change in both precision and recall for each of the three algorithms post fine-tuning and optimisation.

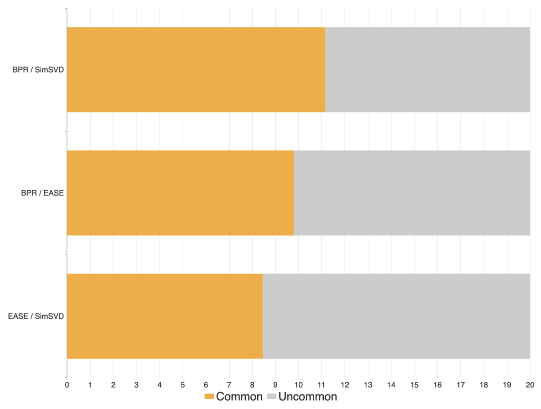

Fig. 5. A graph showing the common and uncommon recommendation rates between the three algorithm pairs.

RQ3: Post-optimisation, which algorithm exhibits the best results overall?
Figure 6 below, shows a comparison between the final post-optimisation precision and recall achieved by each of the three evaluated algorithms. As you can see, the SimSVD algorithm achieved the highest results with a precision of 0.10667 and a recall of 0.34551. The BPR algorithm followed closely behind with a 0.096 precision, however exhibited a significantly worse recall than SimSVD at 0.2589. The EASE algorithm performed very badly when compared with the BPR and SimSVD, with a precision of 0.06042 and a recall of 0.20203. The SimSVD algorithm also exhibited a higher average rating relevance than the other two algorithms, with a value of 3.75099. The BPR algorithm again followed closely behind, with an average rating relevance of 3.7245, while the EASE algorithm once again fell short achieving a 3.52622 average rating relevance.

As you can see, the SimSVD algorithm received the highest results in all evaluation metrics, and outperformed the other two algorithms convincingly on all fronts.

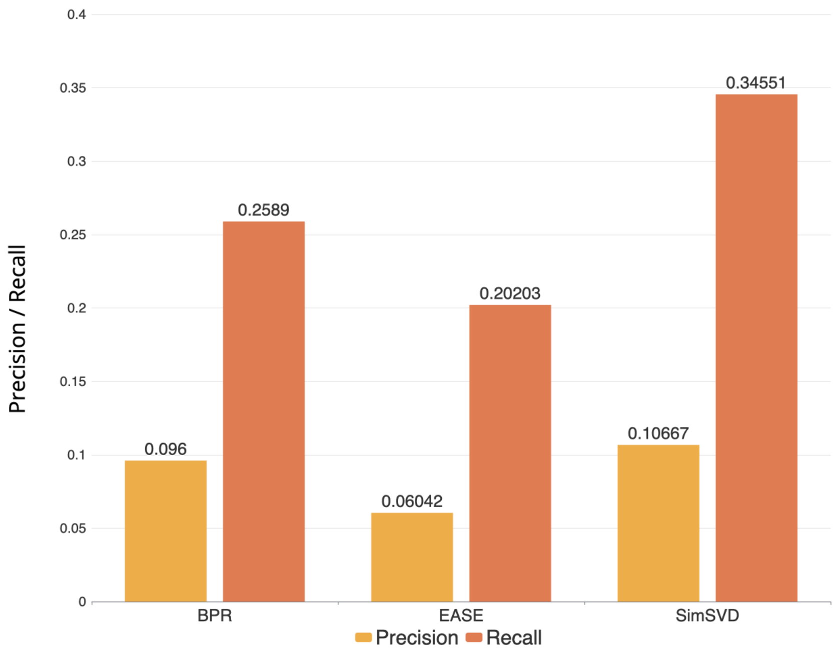

Fig. 6. A graph showing the post-optimisation precision and recall achieved by each of the three algorithms.

4.4 Datasets

For the algorithm analysis section of this report, the *ml20m* movie recommendation dataset was used. This dataset contains separate data files for training data, testing data, tag data, item data and genome scores. For the advanced analysis section of this report, the *'ml100k'* movie recommendation dataset was used. This dataset contains seperate data files for training data, testing data, item data and user data, which allowed me to carry out a user-based analysis on the SimSVD algorithm.

5 Advanced Analysis

Now that the SimSVD item-based collaborative filtering system had been identified as the highest performing algorithm in all evaluation metrics, I decided to try and uncover any potential weaknesses or downfalls within the system.

5.1 Problem Statement

The main potential downfall or weakness that I have decided to analyse for the SimSVD recommender system is the potential for unfairness and bias. Recommender systems generate recommendations through the use of historical data, and algorithmic processes. Some recommender systems are more prone to capturing and even emphasising the biases conveyed in this historical data, and this can lead to unbalanced results wrought with inequalities. For this reason, it is vital to analyse and evaluate the level of bias and unfairness within any potential recommender system.

5.2 Methodology

In order to carry out this evaluation, I used the *'ml100k'* dataset, which unlike the *'ml20m'* dataset, allows for the partitioning of user data dependant on which aspect of the user data is to be analysed. I then modified the *Evaluator* class in my recommender system to allow for the partitioning of user data through custom methods. These custom methods would allow me to single out individuals in the user data dependant on characteristics such as gender, age-range and occupation. Once these users with the desired characteristics are singled out, it is possible to analyse the results obtained from the SimSVD algorithm, and compares these results against results returned from users with other characteristics. An example of this user selection method is the *selectOccupationBasedUserGroup* method below.

```java
private void selectOccupationBasedUserGroup(DatasetReader reader, Integer numUsers,
    String occupation) {

        List<Integer> userGroupList = new ArrayList<Integer>();
        List<Integer> userIds = new ArrayList<Integer>(reader.getUserIds());
        java.util.Collections.shuffle(userIds, new Random(seed));

        if (occupation.equals("A")) {
            numUsers = Math.min(numUsers, userIds.size());
            userGroup = userIds.subList(0, numUsers).toArray(new Integer[numUsers]);
            return;
        }

        int nusers = 0;

        for (Integer userId : userIds) {
            User u = reader.getUser(userId);
            String g = u.getFeatureValue("Occupation");
            if (g == null || g.equals(occupation)) {
                userGroupList.add(userId);
                nusers++;
                if (nusers == numUsers)
                    break;
            }
        }
        userGroup = userGroupList.toArray(new Integer[nusers]);
    }
```

This method can then be called using the following code, which will return the evaluation metrics such as precision, recall, and average relevance for the users with a *'healthcare'* occupation.

```java
Evaluator eval = new Evaluator(alg,reader,N, nusers, "healthcare");
```

5.3 Results and Evaluation

In order to assess the fairness and potential for bias within the SimSVD user-based collaborative filtering system, I decided to analyse three key user characteristics gender, age-range, and occupation. The results and evaluation for this process is as follows.

Gender

In order to test the fairness and bias of recommendations dependant on the gender of the user, I used a method called *selectGenderBasedUserGroup*. This method allowed me to isolate users dependant on whether they where male or female, and allowed me to evaluate and compare the results returned by the SimSVD algorithm dependant on this characteristic.

As you can see in Figure 7 below, there was a relatively wide degree of inconsistency between the standard (both male and female), male-only, and female-only recommendation results. The male-only results exhibited an 18.40% larger precision and a 2.58% larger recall than the female-only results. The male-only results also exhibited a 7.44% higher precision and a 1.18% higher recall than the standard, or dual gender results. This large discrepancy in the recommendation results dependant on gender indicates a large bias and unfairness in the SimSVD recommender system.

A paper by Melchiore, B., et al., titled *"Investigating gender fairness of recommendation algorithms in the music domain"*[4] found that almost all of the algorithms tested in the study had a significant unfairness towards females in terms of precision, recall and coverage metrics. It was also found that some of the

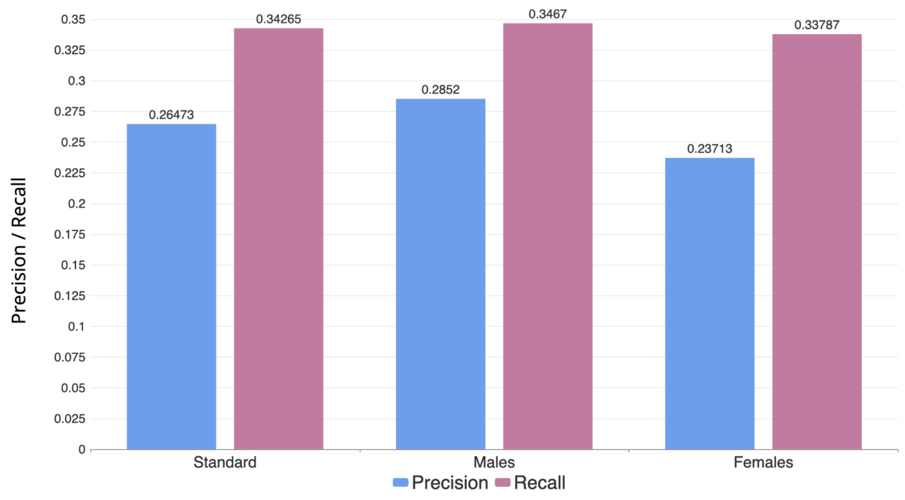

Fig. 7. A graph showing the discrepancies in the precision and recall returned by the standard (both male and female), male and female characteristics.

algorithms that exhibited bias and unfairness actually compounded and made worse the existing biases within the original dataset. In order to evaluate whether the SimSVD algorithm was was creating unfairness, extenuating unfairness, or reducing unfairness, I first had to analyse the *'ml100k'* dataset. Through this analysis, I found that there is a significantly uneven distribution of males and females with a proportion of 71% males and only 29% females in the dataset. I also found that the training data being used by the recommender system is made up of 74.27% male reviews and only 25.73% female reviews, and that the testing data being used is made up of 74.24% male reviews and only 25.76% female reviews.

This massive discrepancy in the proportions of male and female data combined with the gender-based precision and recall results returned by the SimSVD algorithm, shows that the SimSVD item-based collaborative filtering system is not extenuating biases or unfairness present within the dataset, and is actually mitigating them to a small degree. While there is some noticeable unfairness in the results returned by the SimSVD algorithm, the extreme discrepancies in the frequency between male and female data is not reflected in these results.

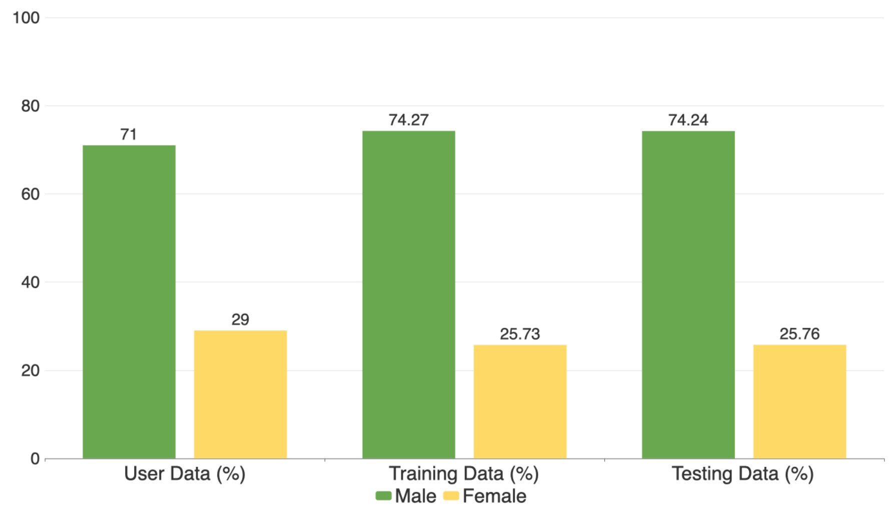

Fig. 8. A graph showing the distribution of male and female data in the vital movie data files.

Age-Range

In order to further test the level of bias and unfairness within the SimSVD item-based collaborative filtering system, I decided to analyse the discrepancies in the results returned to users of different ages. In order to do this, I used a method called *selectAgeBasedUserGroup*, which allowed me to separate users into age-ranges of ten years (e.g 40-49).This then allowed me to evaluate and compare the results returned by the SimSVD algorithm dependant on this characteristic.

As you can see in Figure 9 below, just like gender there is relatively large discrepancies in the results returned by the SimSVD algorithm dependant on the age-range that a user falls into. For example, users aged 40-49 received a 33.32% higher precision, and a 77.96% higher recall than users aged 70-79. This again could indicate large biases and unfairness within the SimSVD algorithm, however just like the gender analysis, it is important to analyse the age data which the recommender is using to produce these results.

In a paper by Neophytou. N., et al, titled *"Revisiting Popularity and Demographic Biases in Recommender Evaluation and Effectiveness"*[5], it was found that significant differences in recommender utility across age brackets is a common weakness and challenge of recommender systems. It was also found that recommender systems can extenuate the age-related biases and unfair aspects of datasets, which is the exact same observation made about gender-based data in the Melchiore, B., et al., study. In order to evaluate wether the SimSVD algorithm was creating unfairness, extenuating unfairness, or reducing unfairness within the age-group recommendation results, I decided to analyse the distribution of user, training and testing data age-ranges.

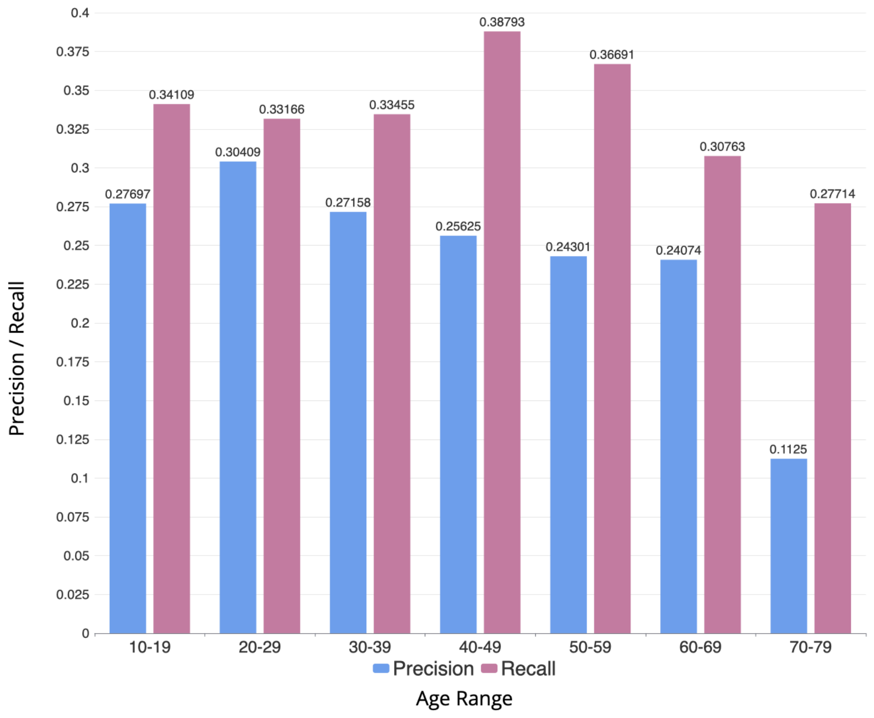

Fig. 9. A graph showing the discrepancies in the precision and recall returned by the age-ranges observed in the movie recommendation dataset.

As shown in Figure 10 below, there are massive inequalities in the representation of age groups within the movie recommendation dataset. Over 35% of each of the three vital data files is comprised of users in the age-range of 20-29, while as little as 0.2% is made up of users in the 70-79 age-range. Users in the age-range of 30-39, and 40-49 are also massively over represented when compared to users in the 10-19 and 60-69 age-range.

This massive discrepancy in the proportions of age-range data combined with the age-based precision and recall results returned by the SimSVD algorithm, again shows that the SimSVD item-based collaborative filtering system is not extenuating biases or unfairness present within the dataset, and is actually mitigating them to a small degree. While again there is some noticeable unfairness in the results returned by the SimSVD algorithm, the extreme discrepancies in the frequency between age-range data is not reflected 1:1 in these results.

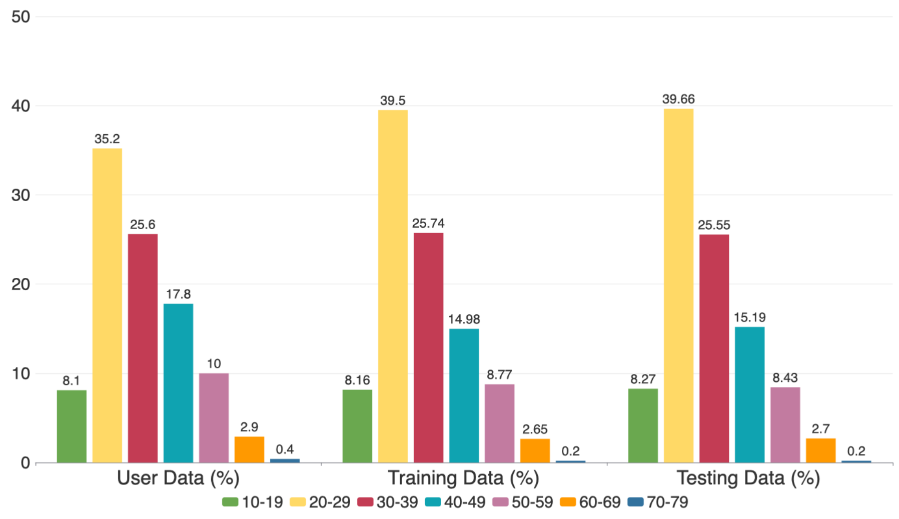

Fig. 10. A graph showing the distribution of age-range data in the vital movie data files.

5.4 Potential Remedies for Bias and Unfairness

There are multiple possible approaches that could be taken in order to mitigate the unfairness and biases exhibited in the above recommender system. The first possible approach is to incorporate the age-range and gender of a user who rated a movie into the movie rating. A user would then be recommended movies based on popularity and average rating, but also the age-range and gender of users who are rating this movie favourably or unfavourably. This would help to mitigate some of the discrepancies observed between users of different genders and different age-ranges, and would ultimately lead to more accurate and precise recommendations.

A paper by Adomavacicius, G., et al., titled *”De-Biasing User Preference Ratings in Recommender Systems”*[6] proposes another approach which could help to mitigate unfairness and biases. In this approach user ratings that are known to be biased are systematically sanitised and neutralised using adjustment rules. This process significantly decreased the rates of bias and unfairness in the recommender system studied by Adomavacicius, G., et al., and I believe that this process would have a significantly positive effect on the SimSVD recommender system discussed above.

Finally, another approach that could be taken is documented in a paper by Zhang, J., titled *”Reducing Recommender Systems Biases: An Investigation of Rating Display Designs”*[7]. In this paper, Zhang successfully reduces the bias and unfairness exhibited in his recommender system by carefully analysing and segmenting the training and testing data provided to the recommender system. Only data which is relevant to a specific user is implemented in training and testing the system, which ultimately led to much more accurate and precise recommendations.

6 Conclusions

In this paper, I have outlined the methodologies of three different recommendation algorithms, Bayesian Personalised Ranking (BPR), Embarrassingly Shallow Auto-Encoders (EASE) and Singular Value Decomposition (SimSVD). I have compared the results achieved by these algorithms on the movie ratings dataset, and have optimised and fine-tuned these algorithms to the highest possible degree. I have also carried out an advanced analysis into the potential for bias and unfairness within the SimSVD recommender system, and have discussed many potential remedies to these biases.

The SimSVD algorithm was the most succesful and accurate algorithm analysed during the course of this paper, and far exceeded the abilities of the BPR and EASE algorithms in the movie recommendation problem. A deep and careful analysis into the potential for bias within the SimSVD system then revealed that while the SimSVD system did exhibit some level of biases and unfairness, it did not extenuate the biases already present within the movie recommendation dataset, and even mitigated them to some degree.

Overall, I am extremely pleased with the results achieved in this paper, and also the discoveries made in the process of obtaining these results. The SimSVD algorithm, is an efficient and accurate way to represent data in order to find similarities and connections between specific data points, which makes it a very useful tool in the recommender systems problem.

References

1. Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. : Paper title. Bpr: bayesian personalized ranking from implicit feedback, (pp. 452–461). Journal, Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence (2009).
2. Dive Into Deep Learning, `https://d2l.ai/chapter_recommender-systems/ranking.html`. Last accessed 07 May 2022
3. Machine Learning Mastery, `https://machinelearningmastery.com/using-singular-value-decomposition-to-build-a-recommender-system/`. Last accessed 07 May 2022
4. Melchiorre, A., Rekabsaz, N., & Brandl, S. : Paper title. Investigating gender fairness of recommendation algorithms in the music domain. Journal, ScienceDirect (2021).
5. Neophytou, N., Mitra, B., & Stinson, C. : Paper title. Revisiting Popularity and Demographic Biases in Recommender Evaluation and Effectiveness. Journal, Microsoft (2017).
6. Adomavicius G., N., Bockstedt, J., & Curley, S. : Paper title. De-Biasing User Preference Ratings in Recommender Systems. Journal, CEUR (2017).
7. Zhang J. : Reducing Recommender Systems Biases: An Investigation of Rating Display Designs. Journal, ResearchGate (2019).