Fake Ratings in Recommender Systems

Adelina Khoroshevskaya Cornelia Gruber Daniel Racek

24th January 2021

Abstract

The Netflix Prize challenge inspired data enthusiasts around the globe to find the best collaborative filtering algorithm for predicting movie ratings. Their findings had and still have a lasting impact on the industry and research community. In this work we analyse the shift in performance of two of the central baseline predictors of the winning solution of this challenge when confronted with fake user ratings, an incident regularly taking place in the real world. Our results show that the predictive performance significantly decreases for the respective item targeted by such fake reviews.

1 Introduction

In 2006, Netflix, Inc., a well-known streaming service company, initiated the Netflix Prize, an open competition to find the best collaborative filtering algorithm to predict user ratings for their movies. For the competition Netflix provided a dataset of over 100 million ratings that 480 thousand users gave to 17,770 movies. Overall the competition lasted for more than three years, at which end the winning team *BellKor's Pragmatic Chaos* won the grand prize of $1,000,000 for improving the previously used Netflix algorithm by more than 10%. [1]

In the most general sense, the idea of collaborative filtering is to make predictions about the interests of a user by collecting preferences from many other users, the underlying assumption being, that if two users have the same preference on one issue, they might have a similar preference on another issue. In terms of predicting ratings in the Netflix Challenge this means that if two users rated several movies similarly, they are more likely to give a similar or equal rating to a completely different movie.

The Netflix Prize had a huge impact on the development and improvement of new and existing algorithms in the field of collaborative filtering, since in order to obtain either milestone prices or the grand price one of the central rules of the challenge was to share the solution and describe it properly in form of a scientific paper [2]. For example matrix factorization, nowadays a well-known class of collaborative filtering algorithms, became widely known during the challenge after a blog post by Simon Funk in 2006 [3]. Moreover, the three papers on the winning solution, which was a joint effort by the teams *BigChaos*, *BellKor* and *Pragmatic Theory*, were cited almost 1000 times [4, 5, 6].

Two central algorithms of the winning solution were k-nearest neighbors (kNN) and Singular Value Decomposition (SVD) for recommendations.[1] Adaptations of those two baseline algorithms were included in abundance in their ensemble solution. For example, just the *Pragmatic Theory* part of the winning solution used over 2000 different instances of kNN algorithms that either differed in their objective or parameterization [5].[2]

The question we are answering in this work is: How do these two baseline algorithms fare once a certain amount of fake ratings is inserted into the dataset? Answering this question is highly relevant, since in the real world such scenarios happen on a regular basis. A fake rating can be described as any rating that is not representing a person's honest and impartial opinion. One can think of paid reviews on Amazon or review bombing on Steam just to name two examples, where such fake reviews regularly take a hold. At the same time variations of these two baseline algorithms are most certainly included in present-day recommendation systems, as the Netflix challenge demonstrated. Therefore, in this work, we design two real-world scenarios, in which we insert fake ratings under different circumstances and compare the predictive performance of the algorithms in various ways before and after.

In the following section we will briefly describe the winning solution and the two central algorithms we want to analyse, SVD and kNN. In section 3 we describe our dataset. Afterwards, in section 4, we

[1] SVD for recommendations is also known as FunkSVD or Funk Matrix Factorization originating from Simon Funk's blog post in 2006.

[2] Differences included the optimization objective, proximity method, prediction method, combinations of some of those methods as well as the parameters given the chosen kNN configuration.

explain the scenarios we are analysing and report the results. Section 5 provides a brief introduction how to deal with fake ratings. Finally, we summarize our findings.

2 Winning solution & Key Algorithms

As mentioned earlier, the winning solution of the Netflix Grand Prize was a joint effort by three teams. Together, they were able to outperform Netflix's original algorithm by 10.06% in terms of RMSE on the test dataset. They achieved this through an ensemble solution, in which they linearly blended together simple predictors with both linear and non-linear blended ensembles that were designed by the different teams.

Ensemble blending was among others achieved through linear, binned linear, polynomial and kernel ridge regression, neural networks, bagged gradient boosted trees as well as SVD feature extraction. Naturally, each ensemble was made up of a variety of different "simple" prediction algorithms. For those, there was a strong focus on matrix factorization techniques, specifically various versions of SVD, since they are computationally very efficient on such a big dataset, but also kNNs, both user & movie based, Restricted Boltzmann Machines (RBMs) as well as multiple different (complex) models to integrate temporal effects, both long and short term.[3]

Since kNN and SVD played such a central role in their solution and hence are most likely included in one way or another in present-day recommendation systems, but are in their baseline form easy-to-understand algorithms, we decided to choose them for our analysis. Although both SVD and kNN are well-known, we want to provide a brief summary on how they work in such a recommendation setting.

K-nearest neighbors in its baseline form is a simple non-parametric memory-based algorithm. A training phase is not required. Instead, when we want to make a prediction, we pick the observation of interest, find the k nearest neighbors in our (training) dataset to our instance based on a proximity method (e.g. correlation based similarity) and then by applying a prediction method (e.g. the mean) on those k neighbors we obtain our prediction. In our recommendation setting we have two options from which perspective we can obtain our prediction, namely from a user or movie perspective.

A possible scenario is the following: we have user u and want to make a prediction on how they rate a movie m (from 1 to 5) using a user-based kNN with k = 3, correlation based similarity and mean prediction. We then take our dataset, calculate the correlation between user u and all other users in our dataset (using their ratings) that have rated movie m, pick the three users with the highest correlation and apply the mean on their ratings for movie m to obtain our prediction. Movie-based kNN works analogously.

Singular Value Decomposition for recommendations (also known as FunkSVD or Funk MF) is a matrix factorization algorithm, in which we factorize the user-movie rating matrix R into a product of two typically lower dimensional matrices, the user matrix Q and the item (movie) matrix P, as shown simplistically in Figure 1.

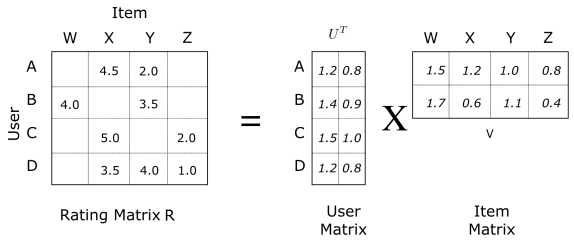

Figure 1: Factorization of the Rating matrix

The former has a row for each user, whereas the latter a column for each item, i.e. movie. The row or column associated with the respective user/item can be interpreted as a vector of latent features that the algorithm tries to learn. The goal is to learn these vectors, such that when we re-produce our rating matrix R through the product of Q and P (see Figure 2) the difference between predicted and actual ratings is minimized.
A predicted rating for a user u and an item i is then given by:

$$\hat{r}_{ui} = q_u^T * p_i$$

[3]One of the key findings in the second year was the "one-day" effect, which might have originated from the fact that multiple users shared the same account.

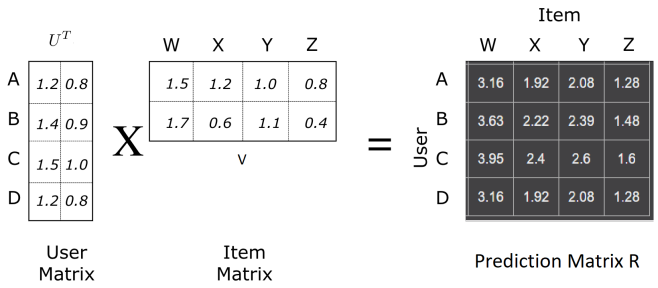

Figure 2: Obtaining the Prediction matrix

While the optimization can be done through a singular value decomposition as known from linear algebra, it is typically done more efficiently through other means, for example using Stochastic Gradient Descent, while additionally adding regularization terms. Predictions are simply obtained after training from the predicted Rating matrix, as seen when comparing Figure 1 with Figure 2. Initially missing ratings are now completed.

A slightly more sophisticated version of the baseline SVD is obtained, when we parameterize the prediction through user and item biases, as well as an overall mean. A predicted rating for a user u and an item i is then obtained through

$$\hat{r}_{ui} = \mu + b_u + b_i + q_u^T * p_i$$

where the overall mean is denoted by μ and the biases for user u and item i by b_u and b_i respectively.

3 Data Description

Since we are interested in the general behaviour of collaborative filtering algorithms when fake ratings are present and not the influence on the Netflix specific movie-user combinations, we chose a benchmark dataset instead of the full Netflix dataset. The benchmark dataset, *MovieLens 100K* [7] enables us to efficiently asses the impact of fake ratings, while having the same structure as the Netflix dataset. Both datasets provide information on users rating movies at a certain point in time. The MovieLens dataset additionally offers basic demographics on the users like their age, gender, occupation and zip-code, as well as the release date or the genre of the movie. There are 100,000 ratings, from 1 to 5, from 943 users on 1682 movies, where each user has at least rated 20 movies.

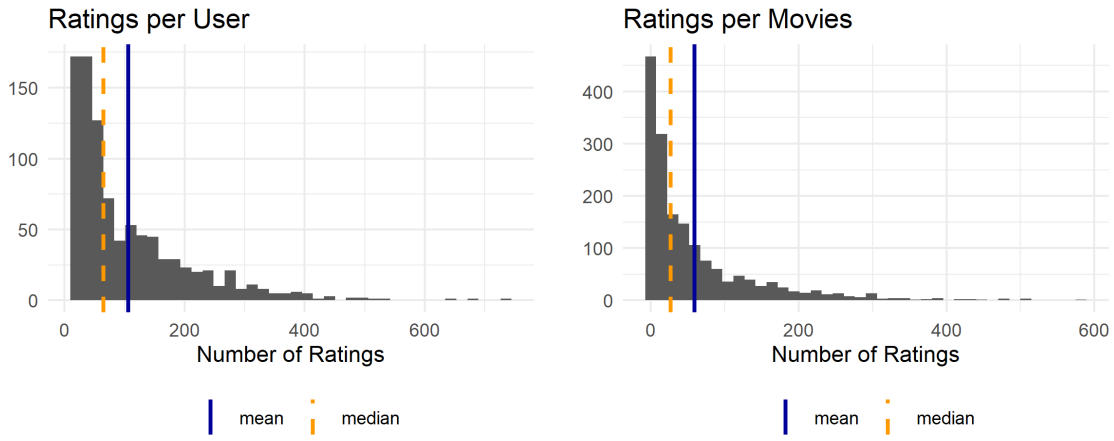

Figure 3: Histogram for the number of ratings per user and per movie

In Figure 3 we can see that most users rate rather a small number of movies, more precisely half of the users rate 65 movies or less. There are however some users with over 600 rated movies. This structure, namely a right skewed distribution with a long tail right of the mean and median, can be seen in the number of ratings each movie got as well. Half of the movies have less than 27 ratings, 75% have less than 80 ratings, whereas the top 5% have over 230 ratings. The average movie has a mean rating of 3.08 while the top 25% have a rating of over 3.65.

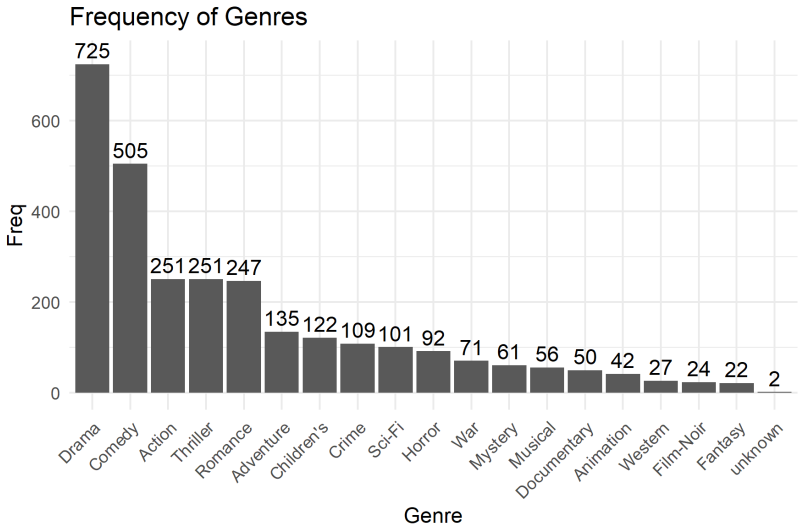

Figure 4: Frequency of movie genres

In Figure 4 we can see that most movies have the genre *Drama*, *Comedy* or *Action* and it is important to note that a movie can have more than one genre. The famous movie "Men in Black (1997)" starring Tommy Lee Jones and Will Smith for example is in the categories *Comedy, Action, Adventure* and *Sci-Fi*. We will insert fake ratings into "Men in Black" since it has many ratings and will therefore affect the prediction for many different users. Additionally, we analyze the influence on "The Lion King (1994)" to see whether predictions for unrelated movies change as well. In particular we want to asses the influence on user 23, a 30 year old, female artist from Ypsilanti in Michigan (close to Detroit) with 151 ratings in total.

When analyzing the user base we find out that 70% of the users are male and mostly young. In Figure 5 we can see that the density of user ages peaks at around 27 and then again at 48. The fact that there are many young users is also reflected in the distribution of occupations, as students make up the largest group.

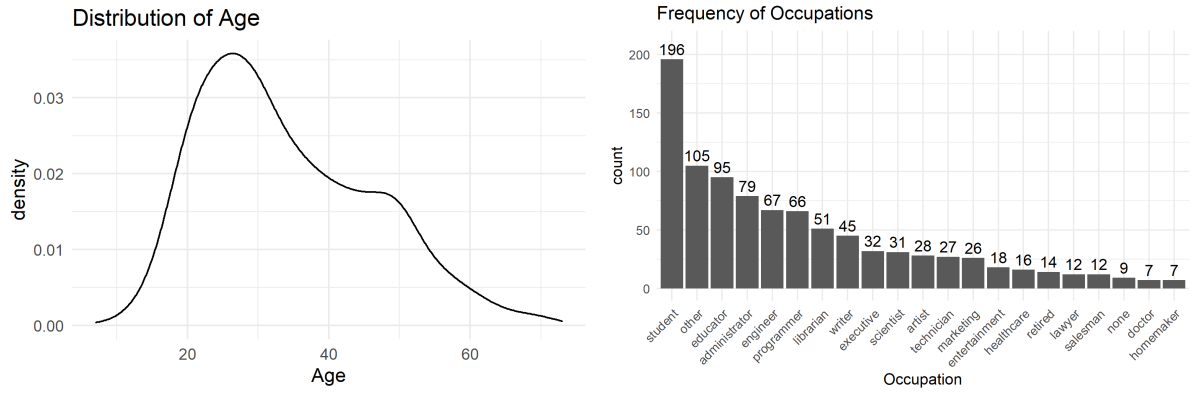

Figure 5: Distribution of Age and Occupation among users

4 Fake Rating Scenarios and Results

We chose two algorithms, namely, kNN and SVD, to test whether adding fake ratings into the training data influences the performance of the models. For the actual implementation we used *surprise*, a package for building and analyzing recommender systems in Python. This package provides four kNN-based algorithms and for each of them a similarity metric as well as the option to choose whether the recommendations should be user or item-based. We tested each available variation of the algorithm with different parameters. A *user-based kNN-baseline with cosine similarity* was able to reach the lowest RMSE with 0.9462 on our test dataset, so we settled for this variant. For SVD the tuning process involved a

grid search among the *number of factors* and *number of epochs* as well as its *biasedness*. Finally, *biased SVD* resulted in the best performance with a RMSE of 0.9268.

Having settled on the algorithm parameters, we also had to fix three components from the data. First, we decided on the actual movie for which we wanted to include fake ratings - *Men in Black*. Second, the unrelated movie - *The Lion King*, which would help us in analyzing how fake ratings affected predictions for other movies. And the third component was the user with *user ID = 23*, on whom we zoomed in to look at the overall change in prediction, as well as the change in predicted ratings for those particular movies.

To actually incorporate fake ratings, we used two strategies. The first scenario resembles review bombing, whereas the second one paid reviews. Review bombing is a phenomenon where large groups of people leave mostly negative reviews in order to harm a specific product. This usually happens when users are unhappy and want to protest against a movie or game itself, for example because of something a developer said or did. Analogously it is also possible to have a positive review bomb, where users want to support a product by giving exceptionally good reviews. For example, this is what happened to the game "Assassin's Creed: Unity" when Ubisoft donated half a million euros to the reconstruction of Notre Dame, after the tragic fire in 2019 [8]. In our first scenario, the movie *Men in Black* will experience such a positive review bomb from existing users and we assess its impact on the rating predictions.

In the second scenario *Men in Black* will also get positive reviews but this time from *new* or *fake* users, i.e. users that did not rate any other movies before. This is unfortunately a common practice especially on websites like Amazon where the purchase decision is influenced by product reviews. Manufacturers pay people for good reviews in order to increase their sales by deceiving potential customers. Paying for fake reviews is also possible for movies in order to get more people to watch them and therefore increase their popularity.

4.1 Scenario 1

For our first scenario we use real users from our training dataset and add fake ratings. The idea was to search for users that did not rate Men in Black yet and then artificially include a 5.0 rating for the movie from those users. At the end, both algorithms, kNN and SVD, were retrained 4 times, once without fake ratings, then with fake ratings ranging from 1%, 10% and 50% of the number of users that did not rate the movie before.

To test the performance, we chose the actual ratings from the test dataset and compared them to the predictions from the algorithms. For the change in RMSE, we used the prediction error for the case without any fake ratings as our baseline.

First, we analyzed the overall mean rating and change in RMSE for both Men in Black and The Lion King. The results are shown in Figure 6 and Figure 7 respectively. Let us first take a look at Men in Black, the movie for which we added the fake ratings, in the context of SVD predictions (see Figure 6a). When we force only 1% of the users to give a 5.0 rating to Men in Black, it does not seem to affect the average predicted rating by much, resulting in a change of only about 0.005 compared to the initial predictions of the algorithm. However, as we increase the amount of fake ratings included in the training data, we clearly see that both the predicted mean rating and the error increase. This increase is particularly evident when we add an additional 50% of 5.0 ratings. In this case, the change in the mean is about 18% and the RMSE increases by almost 21%. Moreover, for the same movie, but now for kNN, this change in predictions is even more pronounced. When half of the users add 5.0 ratings, the mean predicted rating grows by as much as 33.78% and RMSE increases by more than half.

On the other hand, when we look at The Lion King, our unrelated movie, the changes are very minor. For example for SVD, the mean predicted rating only differs by about 0.5% for all of our cases. The change in error is also below 1%. The most interesting difference that can be observed is that the predicted ratings are actually decreasing. One possible reason for this could be that with the artificial increase in popularity of Men in Black, other movies are losing interestingness, which leads to a decrease in their predicted ratings.

After finding out that including fake ratings quite significantly influences predictions for Men in Black, and does not seem to affect The Lion King much, we decided to look at the overall RMSE. That is, how much less accurate our algorithms become on average for all movies. The results are presented in Figure 8a and Figure 8b for SVD and kNN, respectively. Somewhat surprisingly, the percentage change in the error compared to the baseline predictions does not exceed 0.25% for both algorithms. Hence, for most movies the fake ratings seem to have no effect on the performance. Again, kNN performs slightly worse than SVD, and has both a higher RMSE and a higher percentage change in the error.

So far we have seen how the prediction accuracy changes on average when we add fake ratings. Now, we will zoom in onto a single user, the user with *userID = 23* who was examined earlier. As for now, we

% fake ratings	true mean rating	predicted mean rating	% change in rating	RMSE	% change in RMSE
0%	3.5577	3.7137	4.3800	0.9834	0.0000
1%	3.5577	3.7186	4.5200	0.9815	-0.2000
10%	3.5577	3.7912	6.5600	1.0157	3.2829
50%	3.5577	4.1996	18.0400	1.1887	20.8717

(a) SVD

% fake ratings	true mean rating	predicted mean rating	% change in rating	RMSE	% change in RMSE
0%	3.5577	3.7184	4.5200	0.9918	0.0000
1%	3.5577	3.8074	7.0200	1.0143	2.2662
10%	3.5577	4.4168	24.1500	1.3522	36.3337
50%	3.5577	4.7594	33.7800	1.5692	58.2174

(b) kNN

Figure 6: Men in Black overall change in mean rating and RMSE.

% fake ratings	true mean rating	predicted mean rating	% change in rating	RMSE	% change in RMSE
0%	3.8000	3.7815	-0.4900	0.7153	0.0000
1%	3.8000	3.7823	-0.4700	0.7159	0.0894
10%	3.8000	3.7869	-0.3400	0.7187	0.4820
50%	3.8000	3.7872	-0.3400	0.7205	0.7325

(a) SVD

% fake ratings	true mean rating	predicted mean rating	% change in rating	RMSE	% change in RMSE
0%	3.8000	3.7624	-0.9900	0.7591	0.0000
1%	3.8000	3.7626	-0.9800	0.7583	-0.1006
10%	3.8000	3.7650	-0.9200	0.7626	0.4584
50%	3.8000	3.7528	-1.2400	0.7719	1.6879

(b) kNN

Figure 7: The Lion King overall change in mean rating and RMSE

% fake ratings	RMSE	% change in RMSE
0%	0.9262	0.0000
1%	0.9262	0.0049
10%	0.9265	0.0410
50%	0.9271	0.1000

(a) SVD

% fake ratings	RMSE	% change in RMSE
0%	0.9462	0.0000
1%	0.9462	0.0002
10%	0.9475	0.1429
50%	0.9485	0.2461

(b) kNN

Figure 8: Overall change in RMSE

are still focused on our two movies, Men in Black, whose ratings were artificially boosted with differing amounts of 5.0 ratings, and The Lion King, which we consider as our unrelated movie. The results for both, SVD and kNN, can be seen in Figure 9 and Figure 10. First, it is worth mentioning that the baseline predictions for both algorithms, that is, predictions based on training data without fake ratings, were already quite different from the true ratings for both movies. However, this can be attributed to the nature of the data, where we only have integer-valued ratings to derive our information from.

Nevertheless, we can see the same pattern as we have seen before. For both movies, kNN consistently performs worse in terms of predicting the true rating. For example, from Figure 9b one can see that given the true rating for Men in Black by user 23 was 3.0, kNN boosted the prediction to 4.9 when trained on the data with 50% fake ratings included.

One can also see another similar pattern as before, namely, the predicted rating for Men in Black changes much more than that for The Lion King. Given the true rating for the former is 3.0, even the more stable SVD prediction inflates it to approximately 4.1 when trained on data with 50% fake ratings. As for The Lion King and SVD predictions (see Figure 10a), the predicted rating fluctuates around 3.4 and does not change drastically, no matter how many fake ratings we include.

Finally, we will look at the average rating (Figure 11) as well as the prediction RMSE (Figure 12) for our user of interest. Again, the pattern repeats itself: for SVD predictions the mean rating changes at maximum by about 1%, whereas for kNN predictions it is slightly higher, 3.13%. The same holds for RMSE in Figure 12: for SVD maximum percentage increase is only 1.13% and for kNN it is 1.86%.

% fake ratings	true	predicted	% change in rating
0%	3.0000	3.5183	17.2783
1%	3.0000	3.5380	17.9338
10%	3.0000	3.7794	25.9784
50%	3.0000	4.1541	38.4688

(a) SVD

% fake ratings	true	predicted	% change in rating
0%	3.0000	3.7857	26.1898
1%	3.0000	3.9207	30.6898
10%	3.0000	4.6998	56.6611
50%	3.0000	4.9129	63.7621

(b) kNN

Figure 9: Change in rating for user with userID 23 and movie Men in Black

% fake ratings	true	predicted	% change in rating
0%	3.0000	3.4000	13.3322
1%	3.0000	3.3998	13.3283
10%	3.0000	3.4363	14.5428
50%	3.0000	3.4209	14.0289

(a) SVD

% fake ratings	true	predicted	% change in rating
0%	3.0000	3.7412	24.7083
1%	3.0000	3.7413	24.7112
10%	3.0000	3.7378	24.5950
50%	3.0000	3.6927	23.0909

(b) kNN

Figure 10: Change in rating for user with userID 23 and movie The Lion King

% fake ratings	true rating	predicted rating	% change in rating
0%	3.6349	3.6491	0.3901
1%	3.6349	3.6495	0.4012
10%	3.6349	3.6721	1.0237
50%	3.6349	3.6723	1.0289

(a) SVD

% fake ratings	true rating	predicted rating	% change in rating
0%	3.6349	3.7273	2.5415
1%	3.6349	3.7293	2.5971
10%	3.6349	3.7497	3.1575
50%	3.6349	3.7486	3.1276

(b) kNN

Figure 11: Change in rating for user with userID 23

% fake ratings	RMSE	% change in RMSE
0%	0.8467	0.0000
1%	0.8469	0.0246
10%	0.8491	0.2802
50%	0.8563	1.1364

(a) SVD

% fake ratings	RMSE	% change in RMSE
0%	0.8675	0.0000
1%	0.8697	0.2452
10%	0.8872	2.2736
50%	0.8837	1.8638

(b) kNN

Figure 12: Change in RMSE for user with userID 23

Overall, for scenario one, where real users provide fake ratings, we can conclude that incorporating 5.0 ratings for Men in Black has a substantial negative impact on the predictions for this movie, both for a single user and for all users in our dataset on average. Fortunately, as we have seen, it only slightly worsens the predictions for the other movies in our dataset and hence the overall performance of the algorithms. Another important finding from the Scenario 1 is that SVD consistently outperforms kNN both in terms of the accuracy and stability of the predictions.

4.2 Scenario 2

For our second scenario, we modified the training data by appending new users, i.e. users who have not rated anything initially, and then inserted only one 5.0 rating for Men in Black for each of them. Otherwise the procedure was similar to that in Scenario 1: include 1%, 10% and 50% of such new users and compare the performance and error to the baselines.

We will again start our analysis with the overall change in mean and RMSE for all users and the two respective movies, Men in Black (Figure 13) and The Lion King (Figure 14). First off, we can clearly see the difference between the two scenarios in Figure 13a. The percentage increase of the mean rating is about 33% and the increase in RMSE is more than a half for SVD. Whereas kNN demonstrates impressive stability: only about 5% increase in mean rating and less than 0.2% change in RMSE. This situation is the complete opposite of what we had in Scenario 1, where kNN was consistently underperforming compared to SVD. This significant change in performance for kNN can be explained by the fact that we used a *user-based* algorithm, so it did not consider the fake users as neighbors for other users, and therefore did not change its predictions. For The Lion King, we get results quite similar to what we had seen in Scenario 1. Both algorithms seem to be mostly unaffected by the inclusion of fake ratings, and

only slightly change their predictions for The Lion King in the negative direction.

% fake ratings	true mean rating	predicted mean rating	% change in rating	RMSE	% change in RMSE
0%	3.5577	3.7137	4.3800	0.9834	0.0000
1%	3.5577	3.7881	6.4800	1.0198	3.7014
10%	3.5577	4.2733	20.1100	1.2231	24.3746
50%	3.5577	4.7361	33.1200	1.5322	55.8064

(a) SVD

% fake ratings	true mean rating	predicted mean rating	% change in rating	RMSE	% change in RMSE
0%	3.5577	3.7184	4.5200	0.9918	0.0000
1%	3.5577	3.7190	4.5300	0.9919	0.0101
10%	3.5577	3.7234	4.6600	0.9925	0.0706
50%	3.5577	3.7306	4.8600	0.9936	0.1815

(b) kNN

Figure 13: Men in Black overall change in mean rating and RMSE.

% fake ratings	true mean rating	predicted mean rating	% change in rating	RMSE	% change in RMSE
0%	3.8000	3.7815	-0.4900	0.7153	0.0000
1%	3.8000	3.7792	-0.5500	0.7048	-1.4679
10%	3.8000	3.7870	-0.3400	0.7245	1.2862
50%	3.8000	3.7804	-0.5200	0.7168	0.2097

(a) SVD

% fake ratings	true mean rating	predicted mean rating	% change in rating	RMSE	% change in RMSE
0%	3.8000	3.7624	-0.9900	0.7591	0.0000
1%	3.8000	3.7624	-0.9900	0.7591	0.0000
10%	3.8000	3.7625	-0.9900	0.7593	0.0263
50%	3.8000	3.7627	-0.9800	0.7596	0.0659

(b) kNN

Figure 14: The Lion King overall change in mean rating and RMSE

We also analyzed the general change in error for both algorithms, shown in Figure 15a for SVD and Figure 15b for kNN. We can see that when we zoom out and analyze the error for all users and movies on average, both algorithms seem to be performing quite well, with the average RMSE being 0.92 for SVD and 0.94 for kNN, and a slightly more fluctuating SVD error.

% fake ratings	RMSE	% change in RMSE
0%	0.9262	0.0000
1%	0.9262	0.0091
10%	0.9254	-0.0767
50%	0.9280	0.1988

(a) SVD

% fake ratings	RMSE	% change in RMSE
0%	0.9462	0.0000
1%	0.9462	0.0000
10%	0.9462	0.0003
50%	0.9462	0.0018

(b) kNN

Figure 15: Overall change in RMSE

Now let us take a look at our specific user again. Once again, we notice the same contrast to Scenario 1 as before: SVD performing worse than kNN. For example, the former, trained on data with 50% of fake ratings, predicts a rating of almost 4.7 for Men in Black, although the true rating is 3.0 (see Figure 16a). This amounts to a difference of 56%, whereas it is less than half of that for kNN. If we look at the rating change for The Lion King, one can notice that the predicted rating is quite lower for SVD (3.4) than it is for kNN (3.7) for any number of fake ratings. However, this can be explained by the fact that from the start even without any fake ratings included kNN had worse accuracy than SVD. Otherwise, both algorithms seem to be quite stable and do not indicate any dramatic changes in the predicted ratings when including more fake ratings.

Finally, we will look at the overall change in mean and RMSE for our user. Let us first take a look at kNN predictions. In Figure 18b one can see that even though the mean predicted rating differs slightly (by about 2.54%) from the mean, kNN still exhibits good stability for any number of fake users. Same can be seen with the RMSE in Figure 19b, where the percentage increase in RMSE does not exceed 0.04%.

% fake ratings	true rating	predicted rating	% change in rating
0%	3.0000	3.5183	17.2767
1%	3.0000	3.5251	17.5033
10%	3.0000	4.0930	36.4333
50%	3.0000	4.6842	56.1400

(a) SVD

% fake ratings	true rating	predicted rating	% change in rating
0%	3.0000	3.7857	26.1900
1%	3.0000	3.7863	26.2100
10%	3.0000	3.7905	26.3500
50%	3.0000	3.7972	26.5733

(b) kNN

Figure 16: Change in rating for user with userID 23 and movie Men in Black

% fake ratings	true rating	predicted rating	% change in rating
0%	3.0000	3.4000	13.3333
1%	3.0000	3.4322	14.4067
10%	3.0000	3.4126	13.7533
50%	3.0000	3.4185	13.9500

(a) SVD

% fake ratings	true rating	predicted rating	% change in rating
0%	3.0000	3.7412	24.7067
1%	3.0000	3.7413	24.7100
10%	3.0000	3.7419	24.7300
50%	3.0000	3.7430	24.7667

(b) kNN

Figure 17: Change in rating for user with userID 23 and movie The Lion King

If we look at the SVD predictions for our user (Figure 18a), we can see that the algorithm achieves a higher accuracy in terms of the predicted rating, but it is less stable than kNN, with the maximum percentage increase in rating being 1.22%. The same pattern prevails in terms of the error (Figure 19a), where the RMSE for SVD is slightly higher than that of kNN when we include 50% fake users. There, the percentage increase in RMSE is almost 4% compared to 0.04% in kNN, which again underlines the robustness of kNN in this scenario.

% fake ratings	true rating	predicted rating	% change in rating
0%	3.6349	3.6491	0.3901
1%	3.6349	3.6550	0.5535
10%	3.6349	3.6761	1.1339
50%	3.6349	3.6794	1.2233

(a) SVD

% fake ratings	true rating	predicted rating	% change in rating
0%	3.6349	3.7273	2.5415
1%	3.6349	3.7274	2.5436
10%	3.6349	3.7279	2.5586
50%	3.6349	3.7288	2.5834

(b) kNN

Figure 18: Change in rating for user with userID 23

% fake ratings	RMSE	% change in RMSE
0%	0.8467	0.0000
1%	0.8452	-0.1709
10%	0.8590	1.4572
50%	0.8786	3.7709

(a) SVD

% fake ratings	RMSE	% change in RMSE
0%	0.8675	0.0000
1%	0.8675	0.0017
10%	0.8676	0.0143
50%	0.8679	0.0377

(b) kNN

Figure 19: Change in RMSE for user with userID 23

Overall for scenario two, where we added fake users to our training data, we can say that contrary to scenario 1, SVD performs considerably worse than kNN. We still have the same trend, where the accuracy of the predictions is most influenced when we analyze the movie that we added fake ratings for, i.e. Men in Black, while predictions for The Lion King remain generally unaffected. The most interesting result from scenario 2 is the fact that the predictions of kNN remain impressively stable, no matter how many fake ratings we are adding, even for Men in Black.

5 Detection & Countermeasures

So far we have seen how fake ratings can negatively impact algorithms in recommender systems. Hence, dealing with them becomes an essential task, as it can hurt businesses in many different ways. For example in a web-shop, recommending products that a user is not interested in, compared to products that they would be interested in, will ultimately hurt sales in the long run. But what can companies do about such fake ratings?

The most obvious and common strategy is to simply remove them. Naturally, that requires to correctly identify ratings as being fake. For example, a very simple strategy to deal with a specific scenario of fake

ratings, review bombing, could look like the following: Observe the distribution of ratings of an item of interest (e.g. a movie) over time. Once an unusual amount of high or low ratings is detected, e.g. by comparing mean & number of ratings to a weighted average over time, trigger an alert. After such an alert is triggered, a data scientist looks into the issue and decides if a removal of ratings is necessary.

Obviously this simple strategy has many shortcomings. First, it can only deal with one specific scenario of fake ratings. Second, every time an alert is triggered, a data scientist has to explore the issue by hand. A more sophisticated strategy would probably involve designing a classifier that decides for each single rating automatically, if it is real or fake. This could among others include analysing the distribution of ratings of the item, the distribution of ratings given by the user, the text review attached to the rating and many more things. An overview is given in [9]. Concrete strategies are described in [10], [11], [12], [13] and [14].

6 Conclusion

Concluding, we first summarized the winning solution of the Netflix Prize. Then, we highlighted and explained two of the central algorithms, kNN and SVD, playing a central role in the solution. In further consequence, we showed that both are affected by fake ratings, as illustrated by their decrease in performance in our scenarios. Mostly, the performance declines for the respective item/movie, fake ratings are inserted into. In scenario 1, comparable to review bombing, both kNN and SVD are affected, but the latter seems to perform a lot better than the former. On the contrary, in scenario 2, comparable to paid reviews, kNN seems to be hardly affected at all, whereas SVD's performance decreases drastically. With our results we highlight, that detecting and dealing with such fake ratings is absolutely necessary. Finally, we provide a brief introduction and point to further literature on how such measures can be implemented.

References

[1] Netflix, "Netflix prize - congratulations," 2009.
URL: https://www.netflixprize.com/community/topic_1537.html.
Accessed: 26.12.2020.

[2] Netflix, "The netflix prize rules," 2006.
URL: https://www.netflixprize.com/rules.html.
Accessed: 26.12.2020.

[3] Simon Funk, "Netflix update: Try this at home," 2006.
URL: https://sifter.org/simon/journal/20061211.html.
Accessed: 26.12.2020.

[4] Y. Koren, "The bellkor solution to the netflix grand prize," *Netflix prize documentation*, vol. 81, no. 2009, pp. 1–10, 2009.

[5] M. Piotte and M. Chabbert, "The pragmatic theory solution to the netflix grand prize," *Netflix prize documentation*, 2009.

[6] A. Töscher, M. Jahrer, and R. M. Bell, "The bigchaos solution to the netflix grand prize," *Netflix prize documentation*, pp. 1–52, 2009.

[7] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, Dec. 2015.

[8] D. Hartmann, "Steam erlebt so etwas wie positives review-bombing und hinterlässt valve ein wenig unsicher," 2019. Accessed: 21.01.2021.

[9] A. Heydari, M. ali Tavakoli, N. Salim, and Z. Heydari, "Detection of review spam: A survey," *Expert Systems with Applications*, vol. 42, no. 7, pp. 3634–3642, 2015.

[10] L. Akoglu, R. Chandy, and C. Faloutsos, "Opinion fraud detection in online reviews by network effects.," *ICWSM*, vol. 13, no. 2-11, p. 29, 2013.

[11] D. Martens and W. Maalej, "Towards understanding and detecting fake reviews in app stores," *Empirical Software Engineering*, vol. 24, no. 6, pp. 3316–3355, 2019.

[12] A. Heydari, M. Tavakoli, and N. Salim, "Detection of fake opinions using time series," *Expert Systems with Applications*, vol. 58, pp. 83–92, 2016.

[13] S. Shehnepoor, M. Salehi, R. Farahbakhsh, and N. Crespi, "Netspam: A network-based spam detection framework for reviews in online social media," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 7, pp. 1585–1595, 2017.

[14] D. Savage, X. Zhang, X. Yu, P. Chou, and Q. Wang, "Detection of opinion spam based on anomalous rating deviation," *Expert Systems with Applications*, vol. 42, no. 22, pp. 8650–8657, 2015.