Recommender System Optimization through Collaborative Filtering

L.W. Hoogenboom
Econometric Institute of Erasmus University Rotterdam
Bachelor Thesis Business Analytics and Quantitative Marketing

July 2016

Abstract

The amount of available data is growing on a rapid pace. Due to this, data handling is becoming increasingly important. Netflix organised an online competition in 2006 to improve their recommender system. Recommender systems based on collaborative filtering optimize product suggestions based on user preferences by analyzing past user and item rating behaviour. The aim of this research is to optimize movie suggestions by predicting movie ratings based on past ratings. This is done by implementing the neighbourhood based collaborative filtering method described in Bell and Koren (2007), which uses the training data of the Netflix prize competition to train their model. This method differentiates itself by normalizing the data by correcting the ratings for commonly known item and user characteristics. Furthermore, it comes up with an approach which simultaneously derives the interpolation weights for all nearest neighbours. This improves the accuracy of the model in comparison with separately derived interpolation weights. This study adds an extension to this work by correcting for possible noise in the data. This is done in the weight selection, where high interpolation weights are becoming relatively more important than small interpolation weights. Furthermore, the hypothesis that frequently seen movies contain less valuable prediction information is investigated as a second extension. The results show that the addition of both extensions do not improve the prediction performance of the model.

Keywords: *Neighbourhood-based Collaborative Filtering, Algorithm, Case Amplification, Inverse User Frequency.*

Contents

Introduction

Companies are collecting more and more data about consumer behavior. Analyzing this increasing available data is therefore becoming more important. This is the reason why Netflix, an American online movie rental company, organised a competition in order to optimize their movie recommender system (Bennet and Lanning, 2007). Netflix's main goal is to satisfy their customers by providing the movie experience they are looking for, so they stay member for a longer period. This increases their popularity and revenue. To reach this goal they personalize their website by giving personalized movie recommendations for every customer. This research investigates how to optimally give these recommendations using the approach of Bell and Koren (2007).

A qualitative approach to answer this question could be done by hiring movie experts to give standard movie recommendations for several user profiles. Although, it is hard to make these profiles since the data of the user characteristics (like for example: favourite genre and actor) is not available. This research therefore uses a quantitative approach. A recommender system uses past movie evaluations by users (from now on called rankings) to analyse the customer interest in movies.

There are two main ways how these systems work. The first way is *Collaborative Filtering* (CF) which analyses patterns between users and items based on past user preferences. It tries to make automatic predictions about the interests of a user by evaluating past ratings. The main idea behind CF is that if one person has the same opinion about a movie as another person, they are more likely to share the same opinion about other movies, in comparison with randomly chosen others. So, if person 1 and person 2 both like movie A and person 2 likes movie B too, CF recommender systems predict that person 1 appreciates movie B as well.

The second approach is the *content based approach* (CBA). This approach profiles both users and movies and tries to analyse relations between these different profiles. The difference with CF is that CBA uses both user's past behaviour and user and item characteristics, such as age, postcode, movie genre and gender. These characteristics are used to make different user and movie profiles. Unfortunately, the data about these characteristics are not available, so this research does not use the content based approach. Hybrid recommender systems combine the two approaches mentioned above and is proven to be more efficient in some cases (Burke, 2007). Especially when the items include many different elements, the performance of hybrid recommender systems increases since this improves the importance of the textual content. Although, there are no external user nor movie characteristics in the data, so this research uses only CF.

1.1 Literature

One assumption of CF to predict ratings is that missing ratings are missing in a random order. Although, the question remains of how strong this assumption is. Marlin et al. (2012) investigate this issue in a user study in which they collect a random sample of song ratings of users from an online radio service. They find out that the ratings of songs selected by the users themselves have different properties than those of randomly played songs. Beside that, a large number of users also indicate that their opinion of a song does affect whether they rate the song, which clearly violates the assumption that missing ratings are missing randomly. The solution of handling the non-randomness of the data is to add an explicit model of the missing data mechanism. This improves the prediction performance of the random sample ratings significantly.

This research is relevant for this study, since it clearly indicates that one assumption of using CF is not valid. Although, the data description of the provided data lacks information about the possible non-randomness of the data. It is unknown if the ratings are all selected by the users themselves or that they were shown at random. Furthermore, the fact that movies are selected by an independent search of the users themselves or if they watched the movie after a suggestion of Netflix prior to a rating, is also unknown. The complexity of this non-randomness issue combined with the lack of information about the origin of the ratings is the reason why non-randomness is beyond the scope of this research and for that reason ignored.

CF is the most common recommender system if one analyses existing literature on this subject. There are two main CF methods which are memory based CF and model based CF (Su and Khoshgoftaar, 2009). Memory based CF uses the whole user-item database to predict suggestions by finding similar behaving other users which are called neighbours. Bell and Koren (2007) use the *k-Nearest Neighbours* approach which is the most prevalent memory based CF approach. Three major components characterise this approach; finding the most similar users and/or items, data normalization, and determination of the interpolation weights. Bell and Koren's research aims to improve the last two mentioned components. They find out that adjusting for the common features of a ranking data set (the so-called 'global effects') before the normalization of the data improves the root mean squared error (RMSE). They optimise all weights simultaneously using a global optimization, instead of using correlation coefficients to interpolate directly from the neighbours. The advantages of memory based CF is that it is easy to implement, data can be added easily and there is no need of knowledge about the movies. The main challenges are that it is hard to address sparsity and scalability. Furthermore, memory based CF is not able to generate recommendations for new users (Su and Khoshgoftaar, 2009). For websites with a relatively large number of new users and items it would be more efficient to use a

hybrid recommender system.

Model based CF uses models which are able to find difficult patterns based on training data and use these patterns to make predictions for the real data. This method has been developed to solve shortcomings of memory based CF. Model based CF addresses sparsity and scalability problems better than memory based CF. Although, this method has an expensive model building and is for that reason not considered in this research (Su and Khoshgoftaar, 2009).

1.2 Extensions

Bell and Koren (2007) did not adjust their model for noise in the data caused by measurement errors. Although, Netflix's CEO said in an interview with Business Insider reporter McAlone that Netflix is considering to change the five digit ranking system, since it does not work accurately (McAlone, 2016). The main reason behind this is that some users rate the movie according to what they thought about the quality, rather than how they enjoyed the movie, or the other way around. This causes a measurement error in the ratings. Furthermore, some people tend to rate a movie based on their opinion about the current rating given by Netflix, rather than what they actually think of the movie. This leads to the fact that some people who disagree with a very high(low) rating shown by Netflix, tend to rate this movie lower(higher) than their honest opinion to downgrade(upgrade) the movie's rating. This shown rating might influence the customers rating. Furthermore this rating could influence the user's opinion of the movie leading to a violation of the assumption that the ratings are given independently.

An extension is added on the work of Bell and Koren (2007) to handle this noise problem by further investigation of the item weighting. Breese et al. (1998) contains an addition on weight selection in order to decrease the noise in the data. Even more value is attached to big interpolation weights and further decrease the small interpolation weights.

Salton and McGill (1983) found out that popular items are less useful in capturing similarity than less popular items. This effect is partially captured by correcting for the global effect, which adjusts the model for popular and less appreciated movies. Although, this is only the fact that the movie is rated structural higher and not that the movie is rated more often. The last feature is not taken into account in finding similarities in the neighbour selection. This extension can still improve the existing model, since it now both takes the popular effect into account as a global effect and still corrects for the fact that the movie is rated more often in the neighbour selection. This leads to the following research question: Does adjusting the item weights in the neighbour selection with case amplification and inverse user frequency improve the RMSE score of the recommender system generated by Bell and Koren (2007)?

Data

For generating the recommender system, the Netflix Prize dataset is used. This dataset contains over 100 million movie ratings over 480.000 Netflix customers divided over more than 17.000 movies collected between October 1998 and December 2005. The data contains both training data and test data. The training data is used to fit the model and the test data are real ratings which are used to test the prediction performance of the recommender system.

This research uses 10% of the movies and their corresponding ratings. The reason to use a subset of the data is mainly due to the large volume, which results in long computation times. Computations with such large volumes require high memory capacity of computers. Nevertheless, since this research still uses all of the 440.000 corresponding users of the first 10% movies, this subset is still a representative dataset for an online movie rental company which has a smaller movie assortment. Furthermore, if one has a computer with a stronger computation performance and workspace, one could use the whole dataset with the models made in this research.

By analyzing the data, the main conclusion can be drawn that the data is sparse. Around 99% of the user-movie matrix is unknown, so a challenge of this recommender system is how to deal with this sparsity problem.

Methodology

3.1 Normalization of the data

Following the approach by Bell and Koren (2007), the data is normalized by correcting the model for commonly known user, item and user/item-time characteristics, also called *global effects*. Although, neighbourhood CF generates accurate predictions hence it does not consider that certain users may tend to rate lower or higher than other users for example. This is why correcting for these effects before the neighbour selection improves the overall predicting performance of the model.

3.1.1 Explanation global effects

First of all, a brief summary of these commonly known effects will be given. Some movies might get structurally higher ratings than other movies. If one does not correct the ratings before the CF procedure, a popular movie with a large number of low-rated neighbours is likely to be underestimated by the model, since CF does not take this effect into account. Clearly the opposite happens with low-rated movies with many popular neighbours. This is called the *movie effect*. The fact that one movie has neighbours who structurally gave higher or lower ratings, is called the *user effect*.

Besides the fact that the regression model corrects for the the main item and user effects, it also corrects for the global effects. The first four global effects are features about the trend of the ratings of users and movies over time. The $user \times time(user)^{1/2}$ effect corrects for the fact that some users might give higher or lower ratings over time. The $time(user)^{1/2}$ variable is the square root of the elapsed days since the first rating of user u until the date of the rating. The square root of the elapsed days is taken, since exploratory analysis showed that this transformation improved the fit and reduced the RMSE compared to the untransformed number of days (Bell and Koren, 2007). The $user \times time(movie)^{1/2}$ effect adjusts the model for the fact that some users tend to prefer new movies above old movies, or the other way around. It is clear that the movie estimations are also influenced by the elapsed time of users and movies. Therefore, the $movie \times time(movie)^{1/2}$ effect corrects for the fact that some movies are rated higher or lower over time and the $movie \times time(user)^{1/2}$ effect corrects for the happening that some movies are rated higher or lower by users who have been rating for a long time. For example, some documentaries could be more valued by users who have been ranking movies for a longer time.

Other well-known outcomes are the effects of the user and item characteristics on respectively the movies and users. Some users might prefer to watch movies who are well-known and popular among other users, while others prefer to watch special movies which just have been watched by several others. To prevent for overestimates for neighbours who might have seen relatively much high-rated or popular movies, regression of respectively the *user* $\times$ *movie average* and *user* $\times$ *movie support* effects is done to correct the model for these effects. The same adjustments are made for movies. Some movies might be rated by users who tend to rate structurally lower or rate relatively less. The regression of both the *movie* $\times$ *user average* and *movie* $\times$ *user support* adjusts the model for these features.

3.1.2 Sequential regression

The normalization is done by sequential regression. During every step of this procedure one effect is estimated at a time by regressing the residuals r_{iu} of the previous regression on this effect. So only in the first step of the regression the raw ratings are used as dependent variable. For every user(item) effect one estimator per user(item) is generated.

The explanatory variable of item i and user u is the certain global effect denoted by x_{iu} subtracted by its mean for its corresponding item or movie, as shown in equation 1. For the main item and movie effects x_{iu} is equal to one. For simplicity only the methods for estimating the item effects are described. The regression model is shown below and as θ_i the model uses the ordinary unbiased estimator shown in equation 3. Estimating the user effects are done identically by summing over the items instead of over the users in equation 3, where M is the set of users who rated movie i and m is the total amount of users of set M.

$$\hat{x}_{iu} = x_{iu} - \frac{1}{m}\sum_{i \in M} x_{iu} \tag{1}$$

$$new\ residuals = r_{iu} - \theta_i\, \hat{x}_{iu} \tag{2}$$

$$\hat{\theta}_i = \frac{\sum\limits_{i \in M} r_{iu}\, \hat{x}_{iu}}{\sum\limits_{i \in M} \hat{x}_{iu}^2} \tag{3}$$

3.1.3 Sparseness problem estimates

The sparsity of the data could cause that some values of $\hat{\theta}_i$ are based on very few observations. This causes unreliable estimates. To correct for this problem Bell and Koren (2007) avoid overfitting by shrinking individual values of $\hat{\theta}_i$ to a common value. They suppose that θ_i are independent random variables drawn from a normal distribution with known μ and τ^2 as shown in equation 4. Then $\hat{\theta}_i|\theta_i$ is normally distributed with mean θ_i and known σ_i^2.(See equation 5)

$$\theta_i \sim N(\mu, \tau^2) \tag{4}$$

$$\hat{\theta}_i|\theta_i \sim N(\theta_i, \sigma_i^2) \tag{5}$$

They assumed that $\mu = 0$ and σ_i^2 is proportional to $1/n_i$ which resulted in the estimator derived in equation 6, where α is a constant determined by cross validation and n_i is the amount of ratings of movie i.

$$\theta_i = \frac{n_i \hat{\theta}_i}{n_i + \alpha} \tag{6}$$

3.1.4 Cross validation

This research uses cross validation to determine α. Cross validation is a model validation technique, which measures the prediction performance of a model based on training data. It tests the accuracy of the predictions generated by this model for an independent dataset. To determine α in this research, predictions of the test set are generated for several values of α in the range $0 \leq \alpha \leq 100$. The model has the highest prediction accuracy for $\alpha = 50$ according to the RMSE scores shown in appendix A. Therefore, this research uses $\alpha = 50$ for its model.

3.2 Neighbour selection

After the normalization of the data by removing the global effects, the missing ratings have to be estimated. This starts with the neighbour selection. The method discussed in this section does not

require data normalization, so it can be used for any rating data. From now on the ratings mentioned in this section refer to the residuals of the regression from the global effects rather than the raw ratings.

Neighbour selection can be done by investigating item-item or user-user similarities. For big datasets it is efficient to use precomputed similarity matrices to improve the computation time of assigning recommendations. Since the Netflix data contains a lot of movies compared to users, it is both faster and less memory expensive to calculate a movie-movie similarity matrix in this case. Furthermore, Bell and Koren (2007) found that the predictions realised with the item-oriented approach were consistently more accurate. Although, Wang et al. (2006) showed that combining both models can be used for improving prediction accuracy. Note that the user-oriented approach can be done analogous by using the same method as the following item-orientated approach by switching the roles of the movie ratings with the user ratings.

The most common way to choose the most similar neighbours of a movie is by using Pearson correlation. However, Bell and Koren (2007) use an alternative approach which is based on the mean squared error between items. This $m \times m$ item-item similarity matrix, where m is the total amount of movies, provides information on how closely related the movies are and is defined by;

$$s_{ij} = \frac{|U(i,j)|}{\sum\limits_{u \in U(i,j)} (r_{iu} - r_{ju})^2 + \beta}, \tag{7}$$

where $|U(i,j)|$ is the set of common users for movies i and j, r_{iu} is the rating of user u and movie i. The value of β is determined in the same way as the choice of α in the previous section. One calculates the RMSE scores with the normalized data for several values of β and selects the lowest one. These results are shown in appendix B and show that the model has the lowest RMSE for $\beta = 50$. To find the K nearest neighbours for movie i, one chooses the K highest s_{ij} values for given i, which result in a set of neighbours defined by: $N(i;u)$.

3.3 Weight selection

After selecting neighbors $N(i,u)$, one has to calculate the interpolations weights between missing movie i and neighbour j, w_{ij}. For the set of neighbours $N(i;u)$ of every movie i, the weights have to be selected so that they optimize the prediction of r_{iu}:

$$\sum_{j \in N(i,u)} w_{ij} r_{ju} \to \hat{r}_{iu}. \tag{8}$$

The interpolation procedure differentiates itself from others by calculating the weights simultaneously, which accounts for interdependencies among the neighbours. To reach this, Bell and Koren

(2007) consider a hypothetical ideal setting where all neighbours rated every movie in $N(i;u)$. If this setting was correct, the interpolation weights could be easily derived by optimizing a least squares problem denoted by:

$$min \sum_{v \neq u} \left(r_{iv} - \sum_{j \in N(i;u)} w_{ij} r_{jv} \right)^2 \rightarrow \hat{w}. \tag{9}$$

The optimal solution to this least squares problem can be found by solving this system of linear equations. These equations are the same as the result of a linear regression without intercept of r_{iv} on the r_{jv} for $j \in N(i;u)$. This results in the optimal weights given by $Aw = b$, where A is a $K \times K$ matrix and b is a $K \times 1$ column vector defined by:

$$A_{jk} = \sum_{v \neq u} r_{kv} r_{jv}, \tag{10}$$

$$b_j = \sum_{v \neq u} r_{iv} r_{jv}. \tag{11}$$

Additionally, it is very unlikely that users who rated movie i actually rated all the other movies in $N(i;u)$ too. So, if one only evaluates the ratings of users which rated every movie in $N(i;u)$, a lot of information remains unused. Additionally, if there are enough users with complete data for A to be invertible, a significant proportion of the available information about pairwise relationships among ratings by the same user would be ignored. Although, A and b can still be estimated, by dividing equation 10 and 11 by respectively the amount of common raters $|U(i,j)|$ and $|U(j,k)|$. The resulting $K \times K$ matrix $\bar{A}$ has to be estimated for every different neighbour set $N(i;u)$. This makes it time efficient to use a precomputed $m \times m$ matrix with the values of $\bar{A}_{jk}$ for every possible movie pair. This matrix is shown in equation 12. Note that $\bar{b}_j$ is exactly the same formula as $\bar{A}_{jk}$ if $i = k$.

$$\bar{A}_{jk} = \frac{\sum_{v \in U(i,j)} r_{kv} r_{jv}}{|U(i,j)|} \tag{12}$$

Bell and Koren (2007) did even more to correct for the sparseness problem. Some of the averages represented by $\bar{A}_{jk}$ and $\bar{b}_j$ are based on a relatively small amount of users. This could cause the fact that the model describes a random error or noise instead of the underlying relationship between ratings r_{kv} and r_{jv}. This occurrence is called over-fitting. To correct for this contingency and to improve the accuracy of the model, $\bar{A}_{jk}$ and $\bar{b}_j$ are centered towards the mean. This baseline function is the mean of all possible $\bar{A}_{jk}$. To improve the accuracy even more, they distinguish two different averages as baseline. The mean value of the non-diagonal elements of $\bar{A}_{jk}$ is the baseline value for the non-diagonal elements of $\hat{A}_{jk}$ and they shrink the diagonal elements to the average of the diagonal

values of $\bar{A}_{jk}$. The corresponding $K \times K$ matrix $\hat{A}$ and the $K \times 1$ vector $\hat{b}$ are denoted by:

$$\hat{A}_{jk} = \frac{\mu * \gamma + \bar{A}_{jk} |U(j,k)|}{\gamma + |U(j,k)|}, \tag{13}$$

$$\hat{b}_j = \frac{\mu * \gamma + \bar{b}_j |U(i,j)|}{\gamma + |U(i,j)|}. \tag{14}$$

where μ is the corresponding (non-)diagonal average and γ controls the size of shrinkage. This research uses $\gamma = 500$, since this is a typical value when working with residuals of full global effects(Bell and Koren, 2007). The final estimates for A and b are $\hat{A}$ and $\hat{b}$. So the interpolation weights for the ratings of the neighbours of movie i to predict $r_i u$ are given by the solution of the linear system:

$$\hat{A}w - \hat{b} = 0. \tag{15}$$

Notice, that if one works with the residuals of global effects, the effects have to be added back to the predicted $\hat{r}_{iu}$.

3.4 Weight computation

To calculate the weights, one needs to solve equation 15. This can be done by using ordinary linear equations solvers. Although, the weights are more accurate if one solves the equations with the constraint $w \geq 0$, since this avoids redundant overfitting. Bell and Koren (2007) made a quadratic program based on the principles of the Gradient Projection method from Wright and Nocedal (1999). Although, this study uses the Matlab function *quadprog* to compute the interpolation weights, which also solves this quadratic function with non-negative constraints for the weights. This program is both easier to implement and has a decrease in computation time, which makes it beneficial. The function *quadprog* optimizes the optimization function shown in equation 16 and returns weights w, where $\hat{A}$ and $\hat{b}$ are used as input for respectively A and b.

$$\begin{aligned} \underset{w}{\text{minimize}} \quad & w'Aw - 2b'w \\ \text{subject to} \quad & w \geq 0,\ i = 1, \ldots, k. \end{aligned} \tag{16}$$

Extensions

4.1 Case amplification

In this section an extension is added on the work of Bell and Koren (2007) by further investigating the item weighting. Breese et al. (1998) suggest an addition on weight selection which decreases the noise in the data. They attach even more value to big weights and further decrease the small weights with the formula;

$$w_{i,j} = |w_{i,j}|^{\gamma-1} * w_{i,j}, \tag{17}$$

where $w_{i,j}$ is the weight of item i on item j and γ is the case amplification power, where $\gamma \geq 1$. A typical value for the amplification power is 2.5, following Lemire (2005). This study selects γ by comparing the RMSE scores for several γ-values to test if $\gamma = 2.5$ is optimal for this dataset.

4.2 Inverse user frequency

As mentioned in section 1.2, inverse user frequency decreases the weights of movies which are rated relatively often. This corrects for the fact that popular items are bad predictors compared with generally less shown movies. The inverse frequency is defined as:

$$f_i = log\left(\frac{n}{n_i}\right). \tag{18}$$

where n is the total amount of users and n_i is the amount of users which rated item i. This number multiplied by the original predicted item rating $\hat{r}_{iu}$ are the new predicted item ratings $\bar{r}_{iu}$ as shown below.

$$\bar{r}_{iu} = f_i \cdot \hat{r}_{iu} \tag{19}$$

Results

5.1 Generating predictions

Some of the generated rating predictions of the model are greater than five or smaller than one. Since these ratings are not possible in a five star ranking system, one adjusts these ratings to the corresponding boundary. Notice that this adjustment improves the accuracy of the model, since it decreases the RMSE score. Although, valuable information is not taken into account, since a movie with a predicted rating of 5.5 for a certain user has a higher probability to be preferred over a suggestion with an expected rating of 5.0. Therefore, this prediction adjustment is exclusively done to improve the test performance and should not be included in the final prediction model of Netflix.

Some ratings of the test set which have to be predicted do not have any common users in the training set, since this research uses a subset of the Netflix data. It is impossible to make a prediction for these ratings. Although, the aim of this research is to compare the RMSE scores with those of Bell and Koren (2007) and to maintain the reliability of this comparison this study predicts the average of the ratings in the training data for these ratings.

5.2 Calculating prediction performance

To test the prediction performance of the model the RMSE scores are compared with the RMSE scores of Bell and Koren (2007). To calculate these residuals one can use the *Probe* dataset of the Netflix prize competition. This Probe set is a subset of the training data and is included in the dataset to give competitors the opportunity to test their models. This subset only contains the user ID's and movie ID's of around two million ratings. One can calculate the RMSE of a model with the formula shown in equation 20, where P is the set of ratings in the Probe set and r_{iu} refers to a rating which is an element of the Probe set.

$$RMSE = \sqrt{\frac{\sum_{iu \in P} (\hat{r}_{iu} - r_{iu})^2}{\sum_{iu \in P} 1}} \tag{20}$$

The RMSE scores are calculated for three different values of the maximum amount of neighbours K and are shown in Table 1. This research uses different values for K, to test if the results differ significantly. These values are chosen like this, since Bell and Koren (2007) use these values, which makes the results for that reason easier to compare. The results of Bell and Koren (2007) are shown in Table 2.

Table 1: RMSE scores of this research

Data	K = 20	K = 35	K = 50
Raw Data	1.1154	1.1179	1.1220
Double centering	1.0469	1.0490	1.0530
Global effects	1.0645	1.0658	1.0689

Table 2: RMSE scores of Bell and Koren (2007)

Data	K = 20	K = 35	K = 50
Raw Data	0.9536	0.9596	0.9644
Double centering	0.9216	0.9198	0.9197
Global effects	0.9194	0.9179	0.9174

5.3 Comparison of results and analysis

One can see that the RMSE scores of Bell and Koren (2007) are better than those accomplished in this study. This might be caused due to the fact that this research uses a subset of the Netflix data. Even though this subset is representative for the whole dataset, some movies have very few common users. This leads to the fact that some ratings are less accurate than those generated with the whole dataset, which leads to a higher RMSE.

Another explanation for the fact that this research has higher RMSE scores is a possible difference in the used α for the normalization of the data. Bell and Koren (2007) do not mention which value of α they used, nor the used amount of different α-values. If they used cross validation in every step in the sequential regression, it is plausible that they obtained a higher accuracy in their prediction model. As shown in section 3, we used the same α for all the θ_i estimates and did not use cross validation to determine α for every single θ_i. This might have caused the difference in the RMSE scores. However, since α is only used in the denominator of equation 6 it will only have a significant effect on the estimates based on relatively few observations.

It is remarkable that the RMSE scores after the whole normalization are increased compared with the double centering stage. An explanation for this result is that this study does not use cross validation in every sequential regression step to determine the centering constant α. This has caused overfitting in some regression steps of the global effects. Further research, could use cross validation in every sequential regression step to test if these hypothesis is true.

Furthermore, one can see that the differences in RMSE scores among the different K-values in this study are relatively higher than those in Bell and Koren (2007). For every K-value, the RMSE scores of Bell and Koren (2007) are relatively similar, where those of this research decline where K increases. An explanation for this result is the fact that this research only used a subset of the Netflix data. For $K > 20$, some neighbours of movie i might be very unrelated with i. Since the model includes all the 50 most correlated movies, the model includes several neighbours which contain relatively few information about movie i. Bell and Koren (2007) use significantly more movies in their training set, which leads to the fact that the 50 most correlated movies of movie i are still highly correlated with movie i. Moreover, they are relatively more correlated with movie i and for that reason relatively better predictors, than the 50 most correlated movies for movie i in this research.

5.4 Results case amplification

To correct for possible noise in the data case amplification is included in the model. The derived results are shown in Table 3, where all the RMSE scores got higher than those generated without the addition.

Furthermore, the RMSE scores for all the stages in the data normalization are strictly decreasing when the value of γ or K increases. This indicates that there is no noise in the data caused by measurement errors, or that the weight selection method of Bell and Koren (2007) corrects for the noise accurately. Therefore, one can conclude that the addition of case amplification does not improve the accuracy of the model.

Table 3: RMSE scores of case amplification

γ	1.5			2			2.5		
K	20	35	50	20	35	50	20	35	50
Raw Data	1.1214	1.1221	1.1239	1.1249	1.1251	1.1263	1.1262	1.1263	1.1273
Double centering	1.0539	1.0545	1.0563	1.0577	1.0580	1.0592	1.0592	1.0593	1.0603
Global effects	1.0679	1.0682	1.0696	1.0707	1.0708	1.0718	1.0719	1.0719	1.0728

5.5 Results inverse user frequency

The results in Table 4 show that the RMSE scores of the inverse user frequency addition are high. It is especially unexpected for the stages in the normalization before the adjustment of the global effects. This extension adjusts the model for the fact that movies which are watched relatively often can be bad predictors. Since, the amount of ratings per movie is included in the global effect *user* $\times$ *movie support*, one could expect that at least the previous stages would obtain information from the inverse user frequency. However, the amount of ratings per movie is included in the θ_i estimation of every movie effect. Nevertheless, the amount of ratings per movie is already influencing the rating predictions and for that reason this extension causes overfitting. Another explanation for the high RMSE scores is that frequently seen movies are accurate predictors, instead of inaccurate ones.

However, it is remarkable that the RMSE scores of the global effects stage are lower than those of the double centering stage, while the opposite counts for the model without the addition. This could be an indication that frequently watched movies are indeed relatively bad predictors compared with unpopular movies. Although, this is an hypothesis and should be further investigated in other circumstances, since the overfitting causes less reliable results in this study.

Table 4: RMSE scores of inverse user frequency

Data	K = 20	K = 35	K = 50
Raw Data	1.5843	1.5842	1.5843
Double centering	1.5850	1.5850	1.5849
Global effects	1.5751	1.5751	1.570

Inverse user frequency can be combined with case amplification. Table 5 shows that the addition of case amplification to the model with inverse user frequency further increases the RMSE scores for the last two normalization stages. However, this addition improves the estimation results of the model in the raw data stage. This indicates that case amplification can improve the accuracy of a CF model, but that one first has to carefully test the prediction performance of the model before adding this extension.

Table 5: RMSE scores of the combination of both extensions for $\gamma = 1.5$

Data	K = 20	K = 35	K = 50
Raw Data	1.57812	1.57809	1.57809
Double centering	1.61816	1.61815	1.61813
Global effects	1.61857	1.61860	1.61860

Conclusion

The main conclusion of this study is that using a subset of the original data declines the prediction performance of the model significantly. Although, choosing a small amount of neighbours for the individual rating prediction reduces this decline. To answer the research question, adjusting the item weights in the neighbour selection with case amplification and inverse user frequency has not lead to an improvement of the RMSE scores generated by Bell and Koren (2007).

Although, this study used cross validation to determine one constant value to prevent for overfitting in the normalization of the data, instead of deriving multiple constant values. This has lead to inaccurate estimations. The adjustment of this item weighting could have improved the RMSE scores after normalization if these estimations were accurate. This raises the question if this adjustment would have lead to an improvement in the RMSE scores of Bell and Koren (2007) if multiple constant values were derived in the normalization of the data and the whole dataset was used to build the model. Further research can answer this question.

Discussion

This research uses a subset of the original Netflix data. This leads to a decrease in both computation time and storage which eases the difficulty of the methods. Furthermore, it makes the results less comparable with the results of Bell and Koren (2007). Although, this dataset still contains the information about 9 million movies, which still requires efficient programming and data handling.

In section 3.2 the normalized data is used to determine β. However, it would be optimal to calculate the RMSE scores for several values of β, by comparing the predictions based on the original ratings of the training data with the ratings of the test set, since the RMSE scores of the raw rating data are independent for the chosen value of α. However, it turned out that all these RMSE scores where the same for all the chosen β-values. This could be due the fact that this research used a subset of the Netflix data, which caused several non convex optimization problems in the neighbour weight calculation. For that reason we used the normalized data with $\alpha = 50$ to determine β.

The RMSE scores of the inverse user frequency extension are relatively high compared with the other estimation performance results. Therefore, one could challenge the reliability of these estimates and their following conclusions.

As mentioned in section 1.2, the assumption that user ratings are independent might be false. Adding case amplification corrects the model for noise in the data, but might just partially solve the problem that user ratings are dependent on the rating shown by Netflix. Although, correcting for this whole problem is impossible, because the Netflix data does not contain the ratings shown to the users. Moreover, fully correcting for the independency assumption violation is out of the scope of this research.

References

R. M. Bell and Y. Koren. Improved neighborhood-based collaborative filtering. *KDD cup and workshop at the 13th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 7–14, 2007.

J. Bennet and S. Lanning. The netflix prize [www.cs.uic.edu/ liub/kdd-cup-2007/ netflixprize-description.pdf]. 2007.

J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

R. Burke. Hybrid web recommender systems. In *The adaptive web*, pages 377–408. Springer, 2007.

D. Lemire. Scale and translation invariant collaborative filtering systems. *Information Retrieval*, 8(1): 129–150, 2005.

B. Marlin, R. S. Zemel, S. Roweis, and M. Slaney. Collaborative filtering and the missing at random assumption. *arXiv preprint arXiv:1206.5267*, 2012.

N. McAlone. Netflix wants to ditch its 5-star rating system. URL `http://uk.businessinsider.com/netflix-wants-to-ditch-5-star-ratings-2016-1?r=USIR=T`.

G. Salton and M. McGill. Introduction to modern information retrieval. 1983.

X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009:4, 2009.

J. Wang, A. P. De Vries, and M. J. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 501–508. ACM, 2006.

S. Wright and J. Nocedal. Numerical optimization. *Springer Science*, 35:67–68, 1999.

Cross validation in normalization

Table 6: RMSE scores after normalization for $K = 20$ and $\beta = 50$

	$\alpha = 0$	$\alpha = 25$	$\alpha = 50$	$\alpha = 75$	$\alpha = 100$
RMSE	1.2711	1.0780	**1.0645**	1.0646	1.0688

Cross validation in neighbour selection

Table 7: RMSE scores after normalization for $K = 20$ and $\alpha = 50$

	$\beta = 0$	$\beta = 10$	$\beta = 20$	$\beta = 50$
RMSE	1.0799	1.0721	1.0651	**1.0645**