Bootstrapped Collaborative Systems

K. Buchanan

30th August 2018

Abstract

This project implements a bootstrapped recommender system that is designed to match customers to telecommunication products. This is done using collaborative filtering techniques based on k-nearest neighbors (KNN) and matrix factorization using singular value decomposition (SVD). This system allows for product recommendation for customers as well as product prediction that can be used by customer and business services representatives. This project implements a meta approach of bootstrap aggregation in order to improve the stability and accuracy of the systems when working with small datasets. Data involved in this project has been provided by VodafoneZiggo.

Keywords: Recommender System, Collaborative Filtering, k-Nearest Neighbor, Matrix Factorization, Singular Value Decomposition, Bootstrap Aggregating

1 Introduction

Within VodafoneZiggo, there are multiple services/products that a customer can subscribe to; including internet, fixed line, television, and mobile. Currently there is a system called the bundle advice tool (BAT) which will advise customers and customer service representatives to which products would be the best value addition corresponding to the bundling of all services which results in a discount. However, this tool does not help predict which customers would be most likely to add a particular service to their account. For this reason, a system that can evaluate the products a customer currently subscribes to and from this evaluation, produce a list of preferred products, would be a useful tool. This is extremely beneficial for VodafoneZiggo as it finds top products for customers so that sales are increased without reducing the quality of the customer's overall experience. It can also aid company representatives in helping a customer choose which product would be a valued addition to their profile.

Therefore, this issue has been cast to a problem of recommendation. By creating a recommender system, VodafoneZiggo will be able to properly provide a personalized recommendation of which products a customer should add to their profile. This level of personalization attributes to an added degree of customer experience.

In order to do this, a recommender system [3] will be implemented. There are two main types of recommender systems, the first of which being a content-based approach. This approach uses past data in order to build a detailed user profile which includes the ratings of past items [3, 12]. The features of new items are then compared in order to make a recommendation. In contrast, collaborative filtering relies more on the profiles of other users in order to recommend new items [12]. As VodafoneZiggo does not have scores for every customer, building an explicit profile would be an arduous task. For this reason, collaborative filtering techniques will be applied.

Two methods are employed in this project, the first being collaborative filtering based on k-nearest neighbor (KNN) [12]. This method evaluates the ratings customers have given to their products and from this, finds clusters of similar customers. Once these top k-nearest neighbors have been found, KNN makes predictions to which products a customer would prefer to add to their account [8, 11].

The second method employed is collabortive filtering based on matrix factorization [4]. By constructing a table of users and their ratings of products, singular value decomposition can be used to reduce the dimensionality of the matrix, and decompose the original matrix into two three matrices. The product of these matrices is then used to approximate the original data in order to provide predictions on products that have not yet been scored [4].

However, an issue that can arise when using recommender systems that needs to be taken into account is overfitting. This is due to a high generalized complexity due to small amounts of data, where the system knows the training data well, however it cannot generalize beyond this training data. This is an issue for VodafoneZiggo as customers would not be able to properly rely on the system and as a result, would use the recommendation system less. In turn, this creates a loss in data and eventually leads to the system becoming obsolete. Instead, the goal is to refrain from disappointing customers with recommendations so that more customers use the system. This leads to more data and from this, a better system on the next iteration that is less overfitted. This issue of overfitting is especially evident in systems with the problem of cold-start, where a set of user or item scores are not available [2]. The system requires sufficient data to perform well, even if the data is poor. Through collecting more data as the system progresses, the system is improved. When the system is too flexible (for example, when the data is poor), overfitting can become a problem where the variance component of error of the system is quite large [1]. In order to overcome this problem, this project proposes applying bootstrap aggregation to the collaborative systems. Bootstrap aggregation is a meta approach in which given a training set T of size n, bootstrapping finds a new training set $T^{'}$ by sampling of T with replacement [1]. This new training set is then applied to the collaborative system and fit into the model. This is repeated i times and the output is then computed by voting for KNN or averaging for SVD.

Therefore, the problem statement of this project is: *to apply bootstrapping in order to find whether this approach will improve the results obtained by both collaborative filtering techniques.*

From this statement, two research questions have been formulated:

1. Can bootstrapping be applied to recommender systems based on KNN and SVD?
2. If so, is there is an improvement in the results due to the addition of bootstrapping?

By improving the results of the collaborative systems, predictions will be much more accurate allowing VodafoneZiggo to provide better suggestions to their customers. This paper will explore the different works involved in this project in more detail and what methods have been used in order develop these recommender systems.

This paper is organized as follows. Section 2 describes related work. Section 3 will explain the background needed to understand the approach presented in Section 4. Section 5 describes the data used in order to run the experiments presented in Section 6. Section 7 discusses the results of these experiments. Section 8 concludes this report and Section 9 proposes future work.

2 Related Work

In recent work, the use of bootstrapping when working with collaborative systems, is as an initial interview for new users [6, 7, 10]. When new users are introduced into a system, they have not yet provided any item ratings. This is known as the cold-start problem [2] This makes it difficult for the system to provide adequate recommendations. Therefore bootstrapping is used to provide the system with more information about the user.

In [10], the authors propose a “trust-aware bootstrapping”, where new users are invited to give their opinions on other users, rather than on items. Doing so shortens the bootstrapping of the system for new users, and the number of these trust-aware opinions is much less than the number of items.

In [6] and [7], the authors propose to interview users on specifically chosen items or categories, rather than on all items. Again, this shortens the bootstrapping of the system for new users, with the number of carefully chosen items being far less than the total number of items.

Therefore, recent works in bootstrapping collaborative systems have dealt mainly with the problem of introducing new users into the system using bootstrapping. In contrast, this paper will evaluate existing users where overfitting is a large issue, in order to provide more accurate recommendations. The use of bootstrapping is not used for a smaller set of scores or opinions on items, rather it is applied on the collaborative system itself.

3 Background

Recommender systems are tools and techniques that are used in order to provide suggestions of items that are useful or preferred to users, and personalized to that user [4, 12]. These systems can be applied to various fields including which movies to watch, which books to read, and what music to listen to. Since the recommendations retrieved from this system are personalized to each unique customer, this adds another level to the user experience [4]. Therefore many companies have added or are starting to add recommender systems to their user experience; for example, Netflix has integrated their recommender systems directly into their website [4, 12].

3.1 Types of Recommender Systems

As previously mentioned, there are two main types of recommender systems. The content-based approach builds a detailed profile of each user with ratings of past items. It then uses the features of items that were liked in the past in order to recommend new items [3, 12]. Therefore, content-based filtering requires past interests and preferences of a user in order to perform well.

In contrast, the collaborative filtering approach uses the interests and preferences of other users in order to recommend new items [3, 4, 12]. This approach finds similar profiles to the user rather than requiring a detailed profile. The idea behind collaborative filtering is that if two users have rated items similarly in the past, then they will also rate new items similarly.

For this reason, it was decided that collaborative filtering would be the best method for creating a recommender system for VodafoneZiggo as there is not an extensive amount of ratings of past products. Meaning that creating an explicit profile of each user would be too inefficient and not very reliable. As well, it would be a very difficult task to define the features related to past products for these profiles, as product names and what they encompass tend to change often over time. Rather, using collaborative filtering will allow for current ratings to be taken into consideration. Another consideration is since content-based relies upon the user profile to make recommendations, products that are not similar to the customer's current products are unlikely to be recommended. Whereas collaborative filtering will suggest different products as long as a similar customer has used this product and rated it well.

3.2 k-Nearest Neighbor Classifier

The first collaborative filtering technique used, is k-nearest neighbor [12]. As the data contains distinct variables for ratings, k-nearest neighbor system will be based on the classifier.

This nearest neighbor classification follows the idea that for a specific data point, pattern, or vector, similar points can provide useful information [11]. Given a certain data point to be classified, KNN finds the k closest points (neighbors) and then classifies the data point based upon the majority class of the k neighbors [12]. The reasoning behind this classifier is that if a data point is located in a region where a certain class holds majority, it is very likely that this data point is also of this class.

This can be explained mathematically as follows. For a given point $x^{'}$ with class c, and a set $X = \{\{x_1, c_1\}, ..., \{x_n, c_n\}\}$ where x_i is the i-th element in the set and c_i is it's class, KNN looks to find a subset $Y = \{\{y_1, c_1\}, ..., \{y_m, c_m\}\}$ where the distance $\sum_1^m d(x^{'}, y_m)$ is minimal. Thus Y contains those points closest to $x^{'}$, and its class is $c = \text{Majority}(\{c_1, ..., c_m\})$.

This requires a distance function in order to keep the distance between the data point and its k neighbors minimal. As the data uses numeric product scores, Euclidean distance will be used:

$$d(p, q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \tag{1}$$

One of the challenging aspects of KNN is to decide on the value of k [11]. When k is too small, there are only a few points used for neighbors and classification, which makes it sensitive to noise [12]. However, if k is too large the derived subset could contain too many points from other classes.

In this project, KNN will be used as a collaborative filtering technique in which the data point being classified is instead a customer vector containing scores of products. The k value will be determined using internal cross-validation, described further in section 6.2. The system will find the k nearest customers and use these scores in order complete a majority vote to give recommendations.

3.3 Matrix Factorization with Singular Value Decomposition

The second recommender system employed is a collaborative filtering technique based on matrix factorization [4]. This system uses singular value decomposition in order to decompose a matrix of data into two separate matrices. Once this is done, these two matrices can be combined in order to approximate the original matrix [4, 12].

To find these two matrices, singular value decomposition (SVD) is used. This technique finds matrices C and P by minimizing the reconstruction error, where $y_{u,i}$ is a value within the original matrix, and $\boldsymbol{c}_u$ and $\boldsymbol{p}_i$ are the vectors within C and P respectively:

$$err(C,P) = \sum_{u,i}(y_{ui} - \boldsymbol{c}_u \boldsymbol{p}_i^T)^2 \tag{2}$$

Minimizing this error finds $Y = C\Sigma P^T$ which is the SVD of Y, where Σ is a diagonal feature weight matrix. Since Σ is a diagonal matrix, it acts as a scaling mechanism on either C or P, therefore for simplicity purposes, assume that Σ has been merged into one of the two matrices. This gives an approximation, $Y^{'} = CP^T$ [13].

This is done as follows; a mxn matrix Y with m customers and n products is approximated by finding a mxl customer matrix C and a nxl product matrix P, where l is a positive integer lower than n, such that $Y^{'} = CP^T$ approximates Y [12, 13]. This approximation is then used to give recommendations.

4 Bootstrapped Collaborative Systems

As mentioned earlier, an issue involved in recommender systems is the overfitting of data. This is when the generalized complexity of the system is high due to the small amount of data. When a system is too flexible, for example when there is not a sufficient amount of data, the system is not as stable and not as accurate [1]. This is a cause for concern as the system can not generalize past the training data. This causes issues as the recommendations given to a customer are not accurate and error prone. If customers lose faith in such a system, due to inaccurate predictions, they will stop using it. This causes an eventual loss in data and in turn the system can no longer run. In order to overcome this problem, bootstrap aggregation has been added to the two collaborative filtering techniques based on KNN and SVD. Bootstrapping endeavors to lower the flexibility of the system by taking multiple samples of the training data, allowing for replacement [1]. These samples are then used as the input data for the two recommender systems and repeated multiple times. Where each repetition may be overfit itself, by taking the average outcome of the repetitions, a much more stable recommendation is given. As well, since the samples allow for replacement, a degree of flexibility is removed. This in turn lowers the variance of the error and improves the accuracy of the system, as well as helping to avoid overfitting [1].

This meta approach is very useful for VodafoneZiggo, as not all customers have given ratings for current products. This is in line with the concept of the cold-start problem, where scores of customers or products are not available [2]. Therefore the system requires initial data in order to start. In VodafoneZiggo's case, the system is not a true cold-start problem, however the amount of data is poor when new products or new customers are introduced. Bootstrapping is aimed at addressing this issue, while collecting more data and allowing for a better system in the future.

Bootstrapping is applied as follows:

1. Training set T of row length n is sampled n times with replacement creating a new test set $T^{'}$ also of row length n.

2. $T^{'}$ is then used for k-nearest neighbor or singular value decomposition.

3. The resulting predictions of scores are then memorized.

4. This sampling procedure of steps (1) - (3) is completed i number of repetitions, storing all of the outcomes.

5. The outcomes are then voted on in the case of KNN, or averaged in the case of SVD.

6. This average is then used as the recommendation for the customer.

This paper will endeavor to find whether this addition of bootstrapping, will help in the recommendations given by the two collaborative systems.

5 Data

A distinct part of this project was used in order to retrieve the correct data. The difficulty presented itself due to the fairly recent joint venture between Vodafone NL and Ziggo. This caused issues in data extraction as it takes significant time and effort to merge databases of two large companies such as these. Therefore, it was important to meet with many specialists of several databases for both Vodafone NL and Ziggo in order to find the dataset that was ultimately used. As there was not a converged database available at the time of this project, instead a table that is updated monthly with converged data was extracted. This table contains customer profile information as well as which product services the customer currently subscribes to. This data was then manipulated in order to create a two-dimensional matrix that contains a list of products as the columns and the customers as the rows. At first the matrix was filled by entering a 1 where the user currently subscribes to a specific product. However, using this matrix would not properly show customer preference as this solely showed ownership, rather than a score. Therefore an acceptable scoring system needed to be defined.

Within VodafoneZiggo, there is a customer satisfaction indicator called Net Promoter Score (NPS), which each customer gives individually. This score gives the company an overall idea of customer satisfaction in the Netherlands. The score itself is on a scale from zero to ten, where zero indicates low satisfaction and ten shows high satisfaction. From there the NPS is distributed into three classes, where scores from zero to six are considered demoters (customers would outwardly be displeased with the brand), scores of seven and eight are considered neutral (customers are happy, but would not yet actively promote the brand), and scores of nine and ten are considered promoters (customers that would outwardly promote the brand). Therefore, this score was used together with the previous matrix, in order to construct a matrix that contained these scores in place of a one, where the customer subscribes to a product.

6 Experiments

In this section, the experiments of both collaborative systems based on KNN and SVD as well the addition of bootstrap aggregation are explained. The experiments endeavor to show the added value of bootstrapping on KNN and SVD systems.

6.1 Set-up

For these experiments, five datasets are used of varying sizes in order to demonstrate the problem of overfitting. These datasets contain 100, 250, 500, 750, and 1,000 records respectively for 15 products. Each dataset is partitioned into training and test sets and both KNN and SVD techniques are applied. Bootstrapping is then applied using 10, 25, and 50 repetitions. The k value is found using internal 10-fold cross validation and the performance is measured using external 10-fold cross-validation to find accuracy and error of the system.

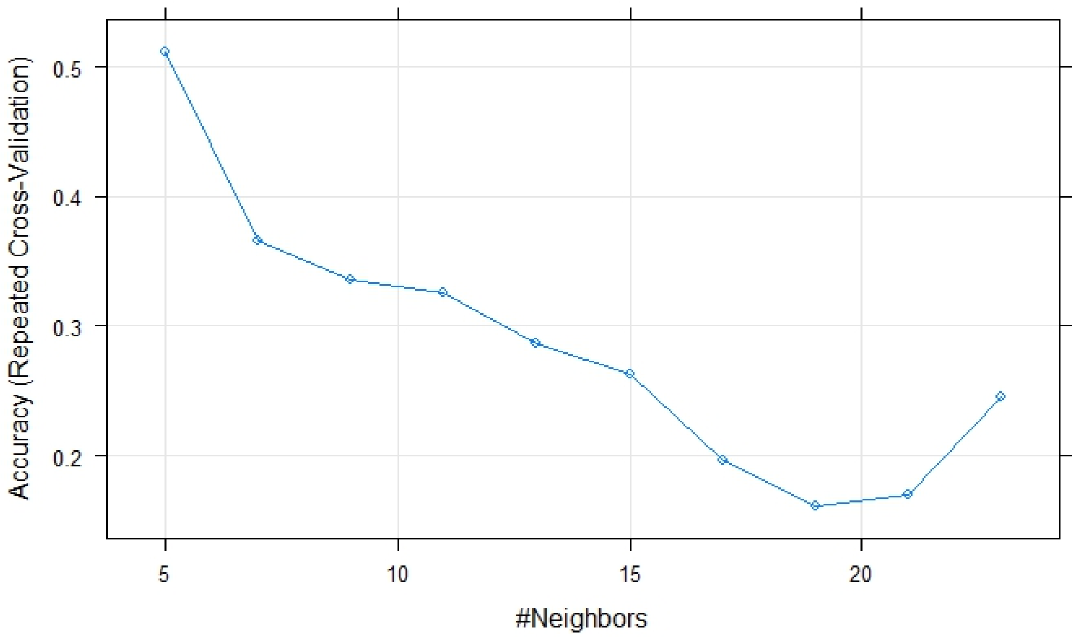
6.2 Results

The k-nearest neighbor collaborative system was run with each of the aforementioned datasets, in order to find predictions for customers. The model was first trained on the training data of each dataset, and the testing data was then used to find predictions. This was then evaluated using 10-fold cross-validation where the dataset was randomly split into ten subsets. For each subset, it is held as a test set where the remaining subsets are used as a training set. The KNN model is then fit on this training set and evaluated on the held-out testing subset [5].

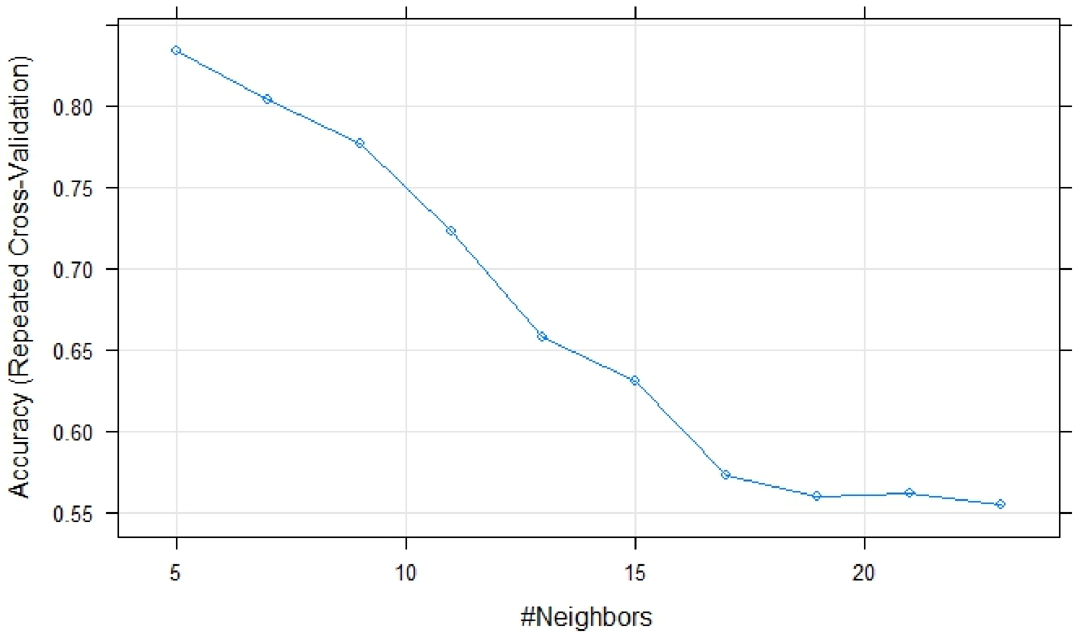
This cross-validation was used in two different aspects. First, k-nearest neighbor requires the value of k to be determined. Secondly, once this k-value has been determined, cross-validation evaluates the accuracy of the KNN collaborative system.

Figure 1a shows the accuracy for the internal 10-fold cross validation of the dataset containing 100 records and figure 1b for 750 records. Therefore, the k value for both datasets were set to 5, as this was the highest accuracy. Similarly all datasets produced a k value of 5 with similar figures.

Once the value of k had been chosen, external cross-validation was then used to find the



(a) The following figure shows the results of internal 10-fold cross-validation on a dataset of 100 records with varying k values. In this instance $k = 5$ is chosen as it shows the highest accuracy of the system.



(b) The following figure shows the results of internal 10-fold cross-validation on a dataset of 750 records with varying k values. In this instance $k = 5$ is chosen as it shows the highest accuracy of the system.

Figure 1: Determining k-value for k-Nearest Neighbor Collaborative System.

	Dataset (number of records)				
Collaborative System	100	250	500	750	1000
k-Nearest Neighbor	51%	75%	81%	88%	90%
With Bootstrapping (10 repetitions)	61%	75%	85%	89%	89%
With Bootstrapping (25 repetitions)	60%	75%	84%	89%	89%
With Bootstrapping (50 repetitions)	60%	74%	84%	89%	89%

Table 1: Accuracy results of k-nearest neighbor collaborative system. The accuracy of the k-nearest neighbor system is shown for each size of dataset, as well as the accuracy of the system with the added approach of bootstrap aggregation with 10, 25, and 50 repetitions.

	Dataset (number of records)				
Collaborative System	100	250	500	750	1000
Singular Value Decomposition	0.4519	0.3162	0.3030	0.3010	0.1899
With Bootstrapping (10 repetitions)	0.3803	0.2740	0.2643	0.2142	0.1899
With Bootstrapping (25 repetitions)	0.3800	0.2881	0.2592	0.2142	0.2046
With Bootstrapping (50 repetitions)	0.3799	0.2716	0.2643	0.2057	0.1923

Table 2: Root squared mean error results (RMSE) of singular value decomposition collaborative system. The RMSE of the singular value decomposition system is shown for each size of dataset, as well as the RMSE of the system with the added approach of bootstrap aggregation with 10, 25, and 50 repetitions.

accuracy of the system using solely KNN and then KNN with bootstrapping applied. Table 1 shows the results of the experiments with k-nearest neighbor on its own, and with the addition of different repetitions of bootstrapping; 10, 25, and 50, respectively.

Next, to study how well bootstrapping improved the results of the singular value decomposition collaborative system, experiments were run with each of the previously mentioned datasets in order to retrieve predictions. It was interesting to see how well the system performed on the larger datasets. For one customer that had rated their internet, fixed line, and mobile service highly at a score of 9 each, the system recommended a similar television product highly at 8.98, rather than recommending other internet, fixed line, or mobile products. External 10-fold cross-validation was then used to evaluate the system and compute root square mean errors (RMSE). Table 2 shows the results of the experiments of singular value decomposition on its own, and with the addition of different repetition of bootstrapping; 10, 25, and 50, respectively.

7 Discussion

From the results of Table 1, it can be seen that the addition of bootstrap aggregation does in fact improve the accuracy of the k-nearest neighbor collaborative system in most circumstances. When the dataset is at its smallest, the amount of overfitting is substantial due to the increase in flexibility, as seen by an accuracy of only 51%. This is improved greatly by the addition of bootstrapping with an increase in accuracy of approximately 10%. As the dataset increases in size, the flexibility of the system reduces, as can be seen in the increase of accuracy over each dataset. The addition of bootstrapping continues to improve the accuracy of the system until the dataset reaches 750 records, however, by a smaller amount each time. This is due to the fact that the flexibility of the system is gradually decreasing and therefore bootstrapping has a smaller effect on the performance.

This improvement in accuracy in the system is very important for VodafoneZiggo, as not all customers have scored their satisfaction for their products. This means that the size of the dataset that can be used, does not always require a large amount of records. Therefore, this bootstrapping technique will greatly improve the quality of predictions and recommendations that are made to customers.

Similarly, Table 2 also shows that the addition of bootstrapping reduces the error of the SVD collaborative system with most datasets. This is especially apparent in smaller datasets as again, the amount of overfitting is large the smaller the dataset. With a RMSE of 0.4519 for the dataset with 100 records, compared to a RMSE of only 0.1899 for the dataset with 1000 records, overfitting is at its highest in the smallest dataset. However, by applying bootstrap aggregation to the system, the RMSE of 0.4519 is reduced to a RMSE of 0.3800. This again will aid VodafoneZiggo in providing more accuracte recommendations to its customers.

Therefore, bootstrapping can be applied to collaborative systems and from experiments with both k-nearest neighbor and singular value decomposition collaborative systems, applying the meta approach of bootstrap aggregation to these systems, allows for an increase in accuracy and a decrease in error of the recommendations predicted. However, it is noted that this increase in reliability of the system is only distinguishable in small datasets. Once a dataset reaches a larger scale, the benefits of bootstrapping fall off. Therefore this technique of adding bootstrapping would be most useful to VodafoneZiggo when starting the recommender system where customer ratings are not yet complete. As the system collects more data, bootstrapping would no longer be as necessary.

8 Conclusion

In conclusion, this paper presents an approach to help combat the problem of overfitting in collaborative systems based on k-nearest neighbor and matrix factorization using singular value decomposition. By applying the meta approach of bootstrap aggregation to both collaborative systems, the accuracy of the system was improved and the error decreased. This was especially evident in smaller datasets where overfitting is at its largest due to the high flexibility of the system. Bootstrapped collaborative systems are able to help reduce this flexibility and improve the quality of recommendations that VodafoneZiggo can provide its customers. Improving recommendations when datasets are smallest, allows VodafoneZiggo to still provide accurate recommendations even when not all customers have provided scores for their products.

9 Future Work

In order to expand on the approach presented in this paper, it would be interesting to apply a reciprocal recommender system based on matrix factorization using singular value decomposition. This would use the matrices found by the matrix factorization in order to discover latent features for both customers and products. These latent features could be attributed to a customer's product necessities, and could be used in order to build a more intelligent recommendation system. This would follow on some of the work presented in [9] regarding student competency discovery.

References

[1] Bootstrap aggregating. Wikipedia, 2018. `https://en.wikipedia.org/wiki/Bootstrap_aggregating`.

[2] Cold start (computing). Wikipedia, 2018. `https://en.wikipedia.org/wiki/Cold_start_(computing)`.

[3] Recommender system. Wikipedia, 2018. `https://en.wikipedia.org/wiki/Recommender_system`.

[4] R. Bell, Y. Koren, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42:30–37, 08 2009.

[5] Jason Brownlee. A gentle introduction to k-fold cross-validation. Machine Learning Mastery, 2018. `https://machinelearningmastery.com/k-fold-cross-validation/`.

[6] Nadav Golbandi, Yehuda Koren, and Ronny Lempel. On bootstrapping recommender systems. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1805–1808, New York, NY, USA, 2010. ACM.

[7] Nadav Golbandi, Yehuda Koren, and Ronny Lempel. Adaptive bootstrapping of recommender systems using decision trees.

In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 595–604, New York, NY, USA, 2011. ACM.

[8] Oliver Kramer. *K-Nearest Neighbors*, pages 13–23. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[9] Evgueni Smirnov Kurt Driessens Mara Houbraken, Chang Sun. Discovering hidden course requirements and student competences from grade data. 2017.

[10] Paolo Massa and Paolo Avesani. Trust-aware bootstrapping of recommender systems. *ECAI 2006 Workshop on Recommender Systems*, pages 29–33, 2006.

[11] Leif E. Peterson. K-nearest neighbor. Scholarpedia, 2009.

[12] Francesco Ricci, Lior Rokach, and Bracha Shapira. *Recommender Systems Handbook*. Springer US, Boston, MA, 2015.

[13] Dmitriy Selivanov. Matrix factorization for recommender systems. Disquis, 2017. `http://dsnotes.com/post/2017-05-28-matrix-factorization-for-recommender-systems/`.