Data Quality Matters in Recommender Systems

Oren Sar Shalom[*]
Microsoft, Israel and
Bar Ilan University, Israel
t-orens@microsoft.com

Shlomo Berkovsky
CSIRO, Australia
shlomo.berkovsky@csiro.au

Royi Ronen
Microsoft, Israel
royir@microsoft.com

Elad Ziklik
Microsoft, USA
eladz@microsoft.com

Amir Amihood[†]
Bar Ilan University, Israel and
Johns Hopkins University
amir@esc.biu.ac.il

ABSTRACT

Although data quality has been recognized as an important factor in the broad information systems research, it has received little attention in recommender systems. Data quality matters are typically addressed in recommenders by ad-hoc cleansing methods, which prune noisy or unreliable records from the data. However, the setting of the cleansing parameters is often done arbitrarily, without thorough consideration of the data characteristics. In this work, we turn to two central data quality problems in recommender systems: sparsity and redundancy. We devise models for setting data-dependent thresholds and sampling levels, and evaluate these using a collection of public and proprietary datasets. We observe that the models accurately predict data cleansing parameters, while having minor effect on the accuracy of the generated recommendations.

1. INTRODUCTION

Data quality is an important practical consideration in many information systems. It can have a strong effect on the performance of the system and the level of user satisfaction. Data quality received significant attention in the general context of information systems, but it has yet to be thoroughly investigated through the recommender systems' prism. For instance, what dimensions of data quality are particularly important for recommenders and what methods can address these? Although there exists some evidence that data quality issues do matter [2, 3, 8], little work has looked into the application of data quality methods to recommenders. These are typically addressed through an ad-hoc data cleansing, such as "prune users with less than X ratings" or "consider data from the recent period Y". But the setting of the data cleansing parameters is often arbitrary and asks for more methodical solutions.

This work addresses two data quality problems in recommender systems. The first refers to the well-established *data sparsity* problem. To this end, we devise a novel model for setting data-dependent threshold for filtering of cold items or users, not having enough data to facilitate generation of reliable recommendations. The second considers the *data redundancy* problem, which may lead to significant overheads at the recommendation model training stages. We propose a method for adaptive sampling of users that can decrease the model training overheads, while still facilitating the construction of accurate recommendation models. This paradigm is used successfully by Azure Machine Learning recommendations [7].

We propose heuristic models for setting the item threshold and user sampling rate, both *without building the recommendation models*. These heuristic models are evaluated using a large collection of public and proprietary recommender system datasets from a range of domains. We observe that the models accurately predict the data cleansing parameters, while having only minor effect on the accuracy of the generated recommendations.

In summary, the contribution of our work is twofold. First, we highlight and demonstrate the importance of data quality matters in recommender systems. Second, we address two practical data quality issues of sparsity and redundancy, by proposing and validating models for adaptive setting of the data cleansing parameters.

2. RELATED WORK

Wang and Strong developed a framework encapsulating the fundamental dimensions of data quality [10]. They derived more than 100 data quality attributes and split these into four dimensions. The *intrinsic quality* dimension refers to the core data characteristics, e.g., accuracy, objectivity, and reputation. *Contextual quality* considers the data in the context of the task at hand and includes attributes like relevancy, completeness, and timeliness. *Representational quality* refers to the format (representation and consistency) and meaning (interpretability) of the data. Finally, the *accessibility* dimension primarily considers data security. Pipino et al. turned to the assessment and metrics of data quality [6]. With the attributes proposed in [10] in mind, they presented a methodology for developing objective metrics communicating the fit of data regardless of the application and task at hand.

Specifically in recommender systems, it has been observed that the rating data can be noisy, imprecise, or outdated [2, 8]. Amatriain et al. demonstrated that offering users to re-rate previously rated items would lead to somewhat different ratings, which substantially change the accuracy of the generated recommendations [2]. Marlin and Zemel questioned the assumption of uniformity in user rating distribution, and showed that this assumption deteriorated the accuracy of collaborative recommendations [4]. Said et al. evaluated the stability of user ratings over time and offered users to re-rate already rated items [8]. It was found that this omnipresent white noise in user ratings poses "the magic barrier" to the accuracy attainable by recommender systems.

[*] This paper is part of the author's PhD thesis.

[†] Partially supported by ISF grant number 571/14.

RecSys'15, September 16–20, 2015, Vienna, Austria.
© 2015 ACM. ISBN 978-1-4503-3692-5/15/09 ...$15.00.
DOI: http://dx.doi.org//2792838.2799670.

To the best of our knowledge, little work has studied the recommenders' data sparsity and redundancy from the data quality perspective. In this work, we systematically address these two data quality attributes and propose methods for dataset-dependent setting of sparsity- and redundancy-related data cleansing parameters.

3. DATA SPARSITY

3.1 Threshold model

Most rating-based recommender system datasets contain a considerable portion of cold users and items. The small number of ratings for these is not sufficient to build a reliable recommendation model, such that the common practice is to omit such items and users outright, as part of the data cleansing process. The real problem, however, is to determine the appropriate cleansing thresholds for a given dataset. A too-low threshold may result in noisy training data and imprecise recommendation models, whereas a too-high threshold may lead to overlooked rating patterns and preclude the system from generating recommendations for these users/items.

A brute-force solution to determining the cleansing threshold could be to exhaustively evaluate all the plausible combinations of item and user thresholds. Even for a small dataset, the number of such combinations is in the thousands, which rules the brute-force solution out for practical business cases. Thus, our aim is to develop a heuristic method that predicts the optimal thresholds for a given user-item rating matrix, without building the model. In this work we focus on the optimization of one – either item or user – threshold, and leave the concurrent optimization of the two for the future. Without the loss of generality, we discuss below the method applied to the item threshold.

We assume that the target threshold value for items is correlated with the average length of the item vectors in the dataset, $\overline{r_i}$, i.e., average number of ratings assigned to an item. However, this feature alone is not sufficient for achieving accurate threshold predictions. Hence, another feature we exploit stems from the parameterization of the power-law distribution of ratings. Let H be the distribution of the item vector lengths. We fit H to a power-law distribution Ax^{-m}, where x is the length of the item vector. Since, typically, there is a small number of popular items with many ratings and a large number of items with a few ratings, m is positive.

We model the item threshold value as a function of $\overline{r_i}$ and m, and assume positive correlation between $\overline{r_i}$ and the item threshold. This is explained by the more robust nature of longer item profiles. Also, we assume negative correlation between the value of m and the threshold. This is due to the observation that when m increases, the weight of the tail of the power-law distribution decreases and there are fewer items with many ratings. Hence, for high m the sparsity of the data is higher and the item threshold is lower. We parameterize the model by a linear multiplier γ. In summary, we model[1] the item threshold IT_d of a dataset d as

$$IT_d = \gamma \cdot \frac{\log(\overline{r_i})}{m^2} \qquad (1)$$

3.2 Evaluation

We use 24 public and proprietary datasets with either implicit or explicit item ratings. Among the public datasets are Movielens, Million Songs, Flixster, Moviepilot, Filmtipset, Yelp, Yahoo! Music (broken down into albums, artists, and tracks), and BookCrossing. The 14 proprietary datasets (referred to as PD) were obtained from various companies and sites, and belong to application domains of eCommerce, car sales, real estate, books, software purchases, grocery shopping, app downloads, video games, and more.

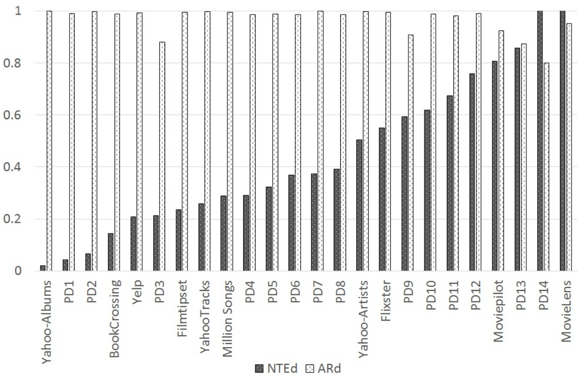

Figure 1: Item threshold predictions of the 24 datasets.

Each dataset was partitioned into the training and test sets using the 90-10 ratio. In order to assess the accuracy of the predictions, we used the *Precision@K* metric [9]. Since the datasets are fairly different, the value of K was set dynamically to 10% of the dataset item set size. The split was repeated ten times, as per the N-fold validation methodology, and the reported precision scores are the averages computed across the ten splits.

We first exhaustively found the optimal item threshold IT_d^{opt} for each dataset d. For this, we gradually increased the value of the item threshold IT, filtered from the data items having less than IT ratings, trained the Matrix Factorization (MF) recommendation model [5, 7] on the cleansed data, and measured the precision score obtained for a fixed test set. The threshold, for which the highest precision was obtained, is referred to as IT_d^{opt}, while the corresponding precision score is P_d^{opt}.

Then, we applied the threshold model in Equation 1 to compute the predicted item threshold IT_d^{pred}. This was done using leave-one-out cross-validation. That is, one dataset d was withheld, the threshold model was trained on the other 23 datasets, and we applied the model to predict the IT_d^{pred} threshold for d. Having set the item threshold to IT_d^{pred}, we trained the recommendation model on the data, with items having less than IT_d^{pred} ratings being filtered out. Given this model, we computed the precision P_d^{pred} of the predictions generated by the model for the fixed test set.

This allows us to derive two performance metrics of the threshold predictions. The first, referred to as the normalized threshold error (NTE), is computed by $NTE_d = |IT_d^{opt} - IT_d^{pred}|/IT_d^{opt}$, and communicates the error of the item threshold predictions. The second quantifies the impact of NTE on the predictions of the recommendation model for the test set. This is referred to as the accuracy ratio (AR) and is computed by $AR_d = P_d^{pred}/P_d^{opt}$. Note that although the threshold model is trained to predict the item threshold IT_d^{pred}, our objective is to cleanse the data in a way that maximizes AR, i.e., $\sum_d(P_d^{pred}/P_d^{opt})$, across the 24 datasets.

We present in Figure 1 the individual NTE and AR scores obtained for the 24 datasets. Each dataset is represented by two bars: the left represents NTE_d and the right – AR_d. The datasets are sorted in an increasing order of NTE_d. As can be seen, the first 14 datasets achieve $NTE_d \leq 0.4$, whereas the next 8 achieve $0.5 \leq NTE_d \leq 1$, and for the last 2 datasets we observe $NTE_d \geq 2$ (these bars are truncated). Overall, the average NTE score across the 24 datasets is 0.632.

[1] We experimented with several other models of IT and the model in Equation 1 yielded the most accurate performance.

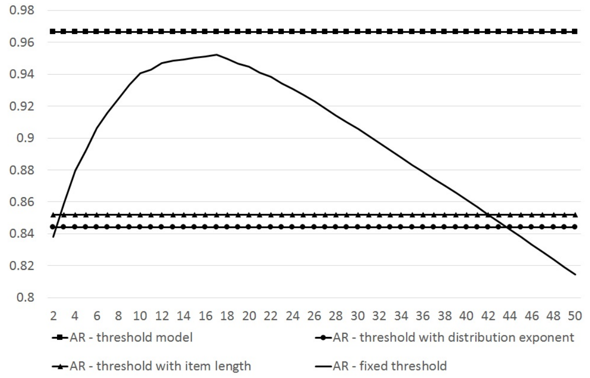

Figure 2: Fixed item threshold experiment.

However, of a greater interest is the impact of NTE on AR. All the datasets demonstrate $AR_d \geq 0.8$, whereas 19 out of the 24 datasets achieve remarkably high $AR_d \geq 0.95$. The overall average AR value across all the datasets is 0.966. Thus, despite the observed threshold prediction errors, the recommendation models built using the cleansed data generate predictions *comparable* to those of the models using the optimal threshold values. We observe correlation of -0.540 between the values of NTE_d and AR_d. This aligns with the intuition that lower errors in the item threshold predictions yield more accurate recommendation models.

To better understand the setting of the item threshold, we carry out another experiment, in which we fix the item threshold IT for all the datasets. We compute the AR, averaged for various values of IT across the 24 datasets. The average AR is compared to three baseline AR using the following IT setting: (i) computed by the model in Equation 1; (ii) computed by a model using only the average item length $\overline{r_i}$; and (iii) computed by a model using only the exponent m of the item length distribution. The results are shown in Figure 2, where the horizontal axis stands for the value of the IT threshold and vertical – for the average AR across the 24 datasets.

As expected, the three baselines are independent of IT and their AR scores are constant. We observe that the model in Equation 1 outperforms the individual models using either $\overline{r_i}$ or m by 13.4% and 14.4%, respectively. The fixed threshold model demonstrates an inverse-curve behavior: for low IT the data is noisy, while for high IT too much data is filtered, such that in both cases the predictions are inaccurate. The highest AR is achieved for $IT = 17$, but this is still 1.5% lower than that of the model in Equation 1. Consider also that such a-priori parameterization may not be feasible for recommenders with dynamic user/item sets and the superiority of our parameter-free model becomes evident.

4. DATA REDUNDANCY

4.1 Model

Another important data quality problem is to identify cases, where some data available in the dataset is redundant, and the recommendation model can be built using a sample of the data. Focusing on random sampling, our target is to pick the lowest sampling rate that will still result in the recommendation model as close as possible to the model that would have been built using the complete data. This is particularly important for practical recommenders and very large scale datasets, where building the complete model may be costly and time consuming. Again, the challenge is to predict the sampling rate, without building the recommendation model.

Unlike in the item cleansing threshold case, there is no optimal sampling rate, because the recommendation model built using the complete data is always superior to, i.e., more accurate than, the one built using the sampled data. Hence, we define the target sampling rate SR as the lowest rate, for which the similarity between the complete recommendation model and the sampled model is greater than a pre-determined parameter Δ. The similarity of the two models is established by comparing the predictions generated by the models for a fixed test set.

In more detail, let us denote by U_d, I_d, and R_d the number of users, items, and ratings, respectively, in a dataset d. We first build the recommendation model using the complete d. As usual in recommender systems, we sample the users in d [1]. Given a sampling rate SR, we retain in the dataset $SR \cdot U_d$ randomly chosen users. Then, we build the recommendation model using the sampled data and generate predictions for a fixed test set. Given a performance metric, we can finally compare the predictions generated by the recommendation model using the sampled data with the ones generated by the model using the complete data.

Since the sampling is done on the users, we assume the level of redundancy to be positively correlated with U_d. We also assume positive correlation with the density of the rating matrix, computed by $\frac{R_d}{U_d \cdot I_d}$. We also incorporate another feature characterising the data, which we denote by *V-structure*. Intuitively, *V-structure* is the relative increase in the similarity of two users given that they have at least one commonly rated item. We compute *V-structure* as the ratio between the average pair-wise similarity of users having at least one jointly rated item and the overall average pair-wise user similarity. Since high *V-structure* of a dataset reflects a greater amount of common rating patterns observed, we posit that the redundancy is positively correlated with *V-structure*.

We model the redundancy level of a dataset d as a combination of three parameters: number of users, density, and *V-structure*. Note that the redundancy is inversely correlated with the sampling rate. That is, when the data is redundant, we sample a small portion of users to build a reliable recommendation model. Also, we need to clamp the sampling rate to the $[0, 1]$ range and keep the function monotonic increasing. We use the hyperbolic tangent function for normalization purposes. In summary, we model[2] the minimal sampling rate SR_d of a dataset d as

$$SR_d = \tanh\left(\frac{1}{\textit{V-structure}_d \cdot \sqrt{U_d} \cdot \frac{R_d}{U_d \cdot I_d}}\right) \qquad (2)$$

4.2 Evaluation

For the evaluation of the sampled models, we used 19 proprietary datasets with implicit and explicit ratings. Each dataset d was partitioned again into the training and test sets using the 90-10 ratio. Also in this experiment the split was repeated ten times and the reported performance was averaged across the ten splits.

We exhaustively found the optimal sampling rate SR_d^{opt} for each dataset d. For this, we first trained MF recommendation model [5, 7] using the complete training dataset and applied this model to generate predictions for a fixed test set. We denote these predictions generated by the model using the complete data as *complete predictions*. Then, we gradually decreased SR by steps of 0.1.[3] For each value of SR we randomly sampled the training data, built the recommendation model using the sampled data, and generated predictions for the same fixed test set.

[2] Here, we also experimented with several other models of SR, and the best performance was achieved by the model in Equation 2.

[3] More fine-grained steps of SR were not sufficiently sensitive.

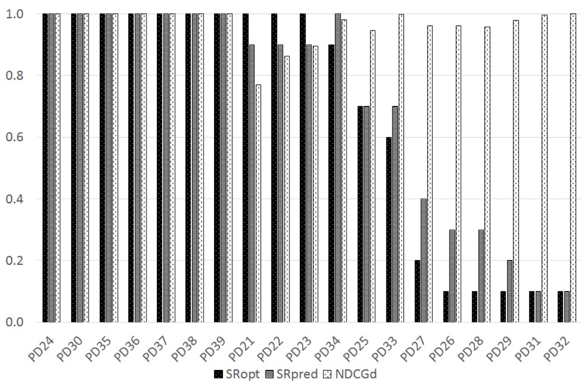

Figure 3: Sampling rate predictions of the 19 datasets.

We used the $NDCG$ metric [9], where the gain of each item was proportional to its rank in the complete predictions, to quantify the predictions generated by the sampled models[4]. We considered the complete model and the sampled model to be sufficiently similar, as long as the $NDCG$ computed by the sampled model for the fixed test set was greater than $\Delta = 0.95$. Thus, we decreased the sample rate SR by steps of 0.1 as long as we managed to obtaine $NDCG \geq 0.95$. The minimal SR, for which this $NDCG$ had been obtained, was considered the optimal sampling rate SR_d^{opt}.

On top of this, we applied the model in Equation 2 to predict the sampling rate SR_d^{pred} for each d. Since the optimal sampling rate SR_d^{opt} was an approximation found by search with steps of 0.1, also SR_d^{pred} was rounded to the closest 0.1 mark. Having set the sampling rate of d to SR_d^{pred}, we created the sampled dataset, then built the sampled MF recommendation model, and generated predictions for the fixed test set. Finally, we evaluated the performance, $NDCG_d^{pred}$, of the recommendation model built using the predicted sampling rate SR_d^{pred}.

Figure 3 presents the results of the sampling rate predictions for the 19 datasets. Each dataset is represented by three bars: namely, SR_d^{opt}, SR_d^{pred}, and $NDCG_d^{pred}$. The datasets are sorted in a decreasing order of SR_d^{opt}. As can be seen, the values of SR_d^{opt} vary across the datasets from 1 (no sampling is needed, all the users are necessary) to 0.1 (only 10% of users are necessary). For 10 datasets out of the 19 we observe $SR_d^{opt} = 1$, which aligns with the established sparsity problem in recommender systems. However, for 6 datasets we observe $SR_d^{opt} \leq 0.2$, indicating that some datasets have high degree of redundancy in the data.

Overall, the predicted sampling rates produced by the model in Equation 2 are close to the optimal ones. We observe that SR_d^{opt} and SR_d^{pred} are identical for 10 datasets out of the 19 (note that for 7 datasets, we observe $SR_d^{opt} = SR_d^{pred} = 1$, i.e., no sampling needed), for 6 datasets the difference is 0.1 (3 over-sampled and 3 under-sampled), and for 6 datasets the difference is 0.2 (SR_d^{pred} over-samples). The average difference between SR_d^{opt} and SR_d^{pred} across the 19 datasets is 0.063. Note that when $SR_d^{opt} \neq SR_d^{pred}$, we prefer to over-sample, i.e., $SR_d^{pred} > SR_d^{opt}$, as in this case, despite keeping unnecessary users, the recommendation model still achieves the desired degree of similarity to the complete model.

We also observe high $NDCG_d^{pred}$ scores, such that for 15 datasets we achieve $NDCG_d^{pred} \geq 0.95$. These include the 7 datasets with $SR_d^{opt} = SR_d^{pred} = 1$, where no sampling is performed and we obviously achieve $NDCG_d^{pred} = 1$. The average $NDCG_d^{pred}$ obtained across the 19 datasets stands at 0.964. Finally, we observe negative correlation of -0.382 between the obtained $NDCG_d^{pred}$ scores and the absolute value of the difference between the predicted and optimal sampling rate, $|SR_d^{opt} - SR_d^{pred}|$. This result is not surprising, since the accuracy of the recommendation models built using the sampled data deteriorates with the error in the sampling rate predictions generated by the model in Equation 2.

5. CONCLUSIONS

Our work was driven by the need to instantiate data quality models for recommender systems. To this end, we addressed two practical considerations of large-scale recommenders: sparsity of user ratings and redundancy of users in the datasets. We developed two models for predicting the data cleansing parameters and demonstrated their validity using a large collection of datasets. Notably, these models capitalize only on the parameters of the datasets and do not require the costly recommendation model building.

This work paves the way for future works on data quality in recommender systems. First, the proposed predictive models for data cleansing parameters were evaluated using the MF recommendation model. However, our models should be evaluated with other recommendation techniques, as, for instance, the item threshold may depend on the underlying recommendation model. Second, the impact of data cleansing on other performance metrics. The filtering of cold users/items and the sampling of users can affect the coverage and the diversity of the generated recommendations. Hence, there is a need to strike the balance between data quality assurance and these metrics. Third, we will consider the ways to incorporate content features of the items and demographic features of the users in the proposed predictive models.

6. REFERENCES

[1] X. Amatriain, A. Jaimes, N. Oliver, and J. M. Pujol. Data mining methods for recommender systems. In *Recommender Syst. Handbook*. 2011.

[2] X. Amatriain, J. M. Pujol, N. Tintarev, and N. Oliver. Rate it again: increasing recommendation accuracy by user re-rating. In *Recommender Syst. Conf.*, pages 173–180, 2009.

[3] S. Berkovsky, T. Kuflik, and F. Ricci. The impact of data obfuscation on the accuracy of collaborative filtering. *Expert Syst. Appl.*, 39(5):5033–5042, 2012.

[4] B. M. Marlin and R. S. Zemel. Collaborative prediction and ranking with non-random missing data. In *Recommender Syst. Conf.*, pages 5–12, 2009.

[5] U. Paquet and N. Koenigstein. One-class collaborative filtering with random graphs. In *Int. World Wide Web Conf.*, pages 999–1008, 2013.

[6] L. L. Pipino, Y. W. Lee, and R. Y. Wang. Data quality assessment. *Commun. of the ACM*, 45(4):211–218, 2002.

[7] R. Ronen, N. Koenigstein, E. Ziklik, M. Sitruk, R. Yaari, and N. Haiby-Weiss. Sage: recommender engine as a cloud service. In *Recommender Syst. Conf.*, pages 475–476, 2013.

[8] A. Said, B. J. Jain, S. Narr, and T. Plumbaum. Users and noise: The magic barrier of recommender systems. In *Int. Conf. on User Modeling, Adaptation, and Personalization*, pages 237–248, 2012.

[9] G. Shani and A. Gunawardana. Evaluating recommendation systems. In *Recommender Syst. Handbook*. 2011.

[10] R. Y. Wang and D. M. Strong. Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems*, pages 5–33, 1996.

[4] As $NDCG$ combines ranking and predictive accuracy metrics, we deem it to be a reliable model performance indicator.