Redundancies in Data and their Effect on the Evaluation of Recommendation Systems: A Case Study on the Amazon Reviews Datasets

Daniel Basaran* Eirini Ntoutsi† Arthur Zimek‡

Abstract

A collection of datasets crawled from Amazon, "Amazon reviews", is popular in the evaluation of recommendation systems. These datasets, however, contain redundancies (duplicated recommendations for variants of certain items). These redundancies went unnoticed in earlier use of these datasets and thus incurred to a certain extent wrong conclusions in the evaluation of algorithms tested on these datasets. We analyze the nature and amount of these redundancies and their impact on the evaluation of recommendation methods. While the general and obvious conclusion is that redundancies should be avoided and datasets should be carefully preprocessed, we observe more specifically that their impact depends on the complexity of the methods. With this work, we also want to raise the awareness of the importance of data quality, model understanding, and appropriate evaluation.

1 Introduction

It is common sense in supervised learning scenarios that a learning algorithm's performance is to be tested on examples that have not been previously seen by the algorithm, i.e., the train and the test sets must be disjoint. Various strategies are used to separate test data and training data, such as a global disjoint split of a given dataset into a training set and a testing set, or randomized methods to repeat the test on several cases or to use more data for training and testing overall (e.g., k-fold cross-validation or bootstrap resampling). If one tests the performance of the model on data that has been used for training, one will tend to *overestimate* the generalization ability of the algorithm.

While this principle is commonly accepted and used in practice, the devil is in the details. Sometimes, datasets and their properties are simply not understood well enough to actually avoid any overlap between training and test sets. This has been discussed, for example, for semi-supervised clustering based on constraints (such as must-links and cannot-links): Pourrajabi et al. [16] show that cross-validation procedures for such semi-supervised tasks can be tricky as information (constraints) used in the test subset can be *implicitly* present (by transitivity of constraints) in the training subset already and thus lead to an underestimation of the error.

Here we discuss the effect of duplicates. If some examples are duplicated, say example A and example B comprise the same information, and happen to be separated, say A is a training example and B is a test example, essentially some algorithm's performance will be tested partly on effectively already seen examples. That such an evaluation should be avoided is immediately clear. However, several algorithms for recommendations have been tested in such a setting simply because the nature of the used datasets, consisting of reviews in different categories of products crawled from the Amazon webpages, was not understood well enough. The popularity of this collection, provided by McAuley and Leskovec [15], can be easily explained: it is huge (approx. 35 millions of reviews), it spans a large period of time (approx. 18 years up to 2013), it is diverse (it covers approx. 30 different product categories) and heterogeneous (user evaluations for certain products come as numerical ratings and as textual reviews).

In this paper, we study the impact of the redundancy in the Amazon reviews datasets on several recommendation methods. The purpose of this study is to bring attention to this problem (that could potentially go unnoticed on other datasets as well), to assess the potential impact of data redundancy on performance evaluations, and in general to raise attention to the importance of dataset preparation and cleaning even in purely academic endeavours. Let us emphasize, though, that having open datasets such as the one released by McAuley and Leskovec, despite imperfections, is an important and appreciated service to the community.

In the remainder of this study, we will first survey how these datasets have been used for evaluation of recommender systems in the literature and describe the methods we use for comparison (Section 2). Then we

*LMU Munich, basaran@cip.ifi.lmu.de
†Leibniz Universität Hannover, ntoutsi@kbs.uni-hannover.de
‡University of Southern Denmark, zimek@imada.sdu.dk

describe the Amazon datasets collection, the nature of redundancy, and our strategy for redundancy elimination as well as the effect of this cleaning on the characteristics of the datasets (Section 3). The main focus of this study is on an extensive experimental comparison of performance assessments with different variants (with and without redundancy) of the Amazon datasets (Section 4) and a discussion of our results and their implications for the literature on recommendation systems (Section 5). We conclude with summarizing our findings and with discussing consequences as well as related issues in other fields of data mining (Section 6).

2 Related work

A recommender system [18] consists typically of a set of items I, a set of users U and the (numerical) ratings of users for certain items, i.e., $r_{u,i}$, where $u \in U$ and $i \in I$. Typically, the cardinality of the item set I is very high and users rate only a few items. For an item $i \in I$ unrated by a user $u \in U$, i.e., $r_{u,i} = \varnothing$, the recommender estimates a relevance score $rec(u, i)$ of the potential interest of user u for item i.

2.1 The Amazon reviews data in the literature Since their publication in 2013 [15], the collection of Amazon reviews datasets has been used widely for evaluating recommendation techniques. According to Google Scholar there are currently 298 citations (last checked: October 14, 2016) to the original paper [15], where McAuley and Lescovic use the collection to evaluate their approach that combines latent rating dimensions (learned from the numerical ratings) with latent review topics (learned from the reviews). Later examples using the datasets include Johnson and Zhang [8] using the "Electronics" dataset in their work on convolutional neural networks for text categorization. McAuley et al. [13] trained their algorithm on image-based recommendations on styles and substyles on several datasets. McAuley et al. [12] use the larger sets to infer networks of substitutable and complementary products, which recommend suitable shoes for a specific outfit. McAuley et al. [11] use the "Toys" dataset to learn attitudes and attributes from multi-aspect reviews. Yang et al. [21] use parts of some datasets in their work on predicting the helpfulness of review texts. McAuley and Leskovec [14] use certain datasets of the collection, namely the "Fine Foods" and "Movies" datasets, to evaluate their method that models user personal evolution or experience for recommendations. Lim et al. [10] used a subset of the "Video Games" dataset to evaluate their algorithm on top-n recommendations with missing implicit feedback.

2.2 Methods used for this case study We include the method of McAuley and Lescovec [15] that introduced the collection, but also traditional recommendation methods:

Offset Model predicts the rating of a user u for an item i by an offset term μ equal to the average across all ratings in the training set:

$$rec(u, i) = \mu \tag{2.1}$$

This is a baseline method to predict a rating.

Offset with Bias Model corrects the offset model for the bias of users and items by adding the average ratings of the query user (user bias b_u) and the average ratings for the recommended item (item bias b_i) [3]:

$$rec(u, i) = \mu + b_u + b_i \tag{2.2}$$

User and item biases model the preference in rating of a user and an item, respectively. For example, some users tend to give higher ratings than other users.

Latent Factors Model is the standard model: the rating of a user u for an item i is modeled as the sum of the average score, item and user biases, and the similarity of users and items in the latent space:

$$rec(u, i) = \mu + b_u + b_i + \gamma_u \cdot \gamma_i \tag{2.3}$$

γ_u and γ_i are user and item factors modeling the preferences of users and the properties of items, respectively.

Matrix Factorization Model [17, 5] is very similar to the Latent Factors Model, approximating unobserved ratings of user u for an item i by combining a user specific feature matrix U and an item specific feature matrix I. Each user and each item is assigned k features which are learned by Stochastic Gradient Descent. The relevance of an item i for a user u is given by:

$$rec(u, i) = \langle U_u, I_i \rangle = \sum_{n=1}^{k} U_{u,n} I_{i,n} \tag{2.4}$$

Biased Matrix Factorization Model [17, 5] adds user and item biases to the Matrix Factorization model:

$$rec(u, i) = b_u + b_i + \sum_{n=1}^{k} U_{u,n} I_{i,n} \tag{2.5}$$

Hidden Factors Model [15] combines numerical ratings with textual reviews by linking latent factors in review ratings (discovered through the Latent Factors model) to hidden factors in review texts (discovered through Latent Dirichlet Allocation (LDA)). In particular, it discovers topics that are correlated with the hidden factors of products and users, i.e., γ_i and γ_u, respectively.

Table 1: Amazon reviews datasets

Dataset	#Users	#Items	#Reviews
Amazon Instant Video	312,930	22,204	717,651
Arts	24,071	4,211	27,980
Automotive	133,256	47,577	188,728
Baby	123,837	6,962	184,887
Beauty	167,725	29,004	252,056
Books	2,588,991	929,264	12,886,488
Cellphones&Accessories	68,041	7,438	78,930
Clothing&Accessories	128,794	66,370	581,933
Electronics	811,034	82,067	1,241,778
Gourmet Foods	112,544	23,476	154,635
Health	311,636	39,539	428,781
Home&Kitchen	644,509	79,006	991,794
Industrial&Scientific	29,590	22,622	137,042
Jewelry	40,594	18,794	58,621
Kindle Store	116,191	4,372	160,793
Movies&TV	1,224,267	212,836	7,850,072
Music	1,134,684	556,814	6,396,350
Musical Instruments	67,007	14,182	85,405
Office Products	110,472	14,224	138,084
Patio	166,832	19,531	206,250
Pet Supplies	160,496	17,523	217,170
Shoes	73,590	48,410	389,877
Software	68,464	11,234	95,084
Sports&Outdoors	329,232	68,293	510,991
Tools&Home Improvem.	283,514	48,059	409,499
Toys&Games	290,713	53,600	435,996
Video Games	228,570	21,025	463,669
Watches	62,041	10,318	68,356

We thus have 6 recommendation methods representative for different levels of complexity. We will see in the experiments that a method's sensitivity to redundancy appears to be stronger if the model is more complex and might have a stronger tendency to overfit.

3 The Amazon reviews datasets

The Amazon Reviews datasets were collected from the Amazon webpages by McAuley and Leskovec [15] and were made publicly available.[1] There are approx. 35 million reviews in the collection spanning a period of 18 years, up to March 2013. The reviews fall into different categories like books, movies, clothing etc., treated as separate datasets in the evaluation of methods. In addition to the review text, a numerical rating of the helpfulness of the review is given, characterizing how much other users valued the review. Furthermore the datasets provide related data like what products were bought in combination, what products were bought after viewing, what products a user bought and what the user viewed, and the sales ranks of products. A detailed description of the categories and the amount of data per category, namely the number of items, users and reviews, is provided in Table 1.[2]

Table 2: Examples of duplicated reviews (and the result of duplicate elimination).

User	Product	Time	Review	Rating
user 1	item 1	t_1	"As a professional plumber..."	3
~~user 1~~	~~item 2~~	~~t_1~~	~~"As a professional plumber..."~~	~~3~~
~~user 1~~	~~item 3~~	~~t_1~~	~~"As a professional plumber..."~~	~~3~~
user 3	item 8	t_2	"This is a great product..."	5
user 3	item 7	t_3	"This is a great product..."	5

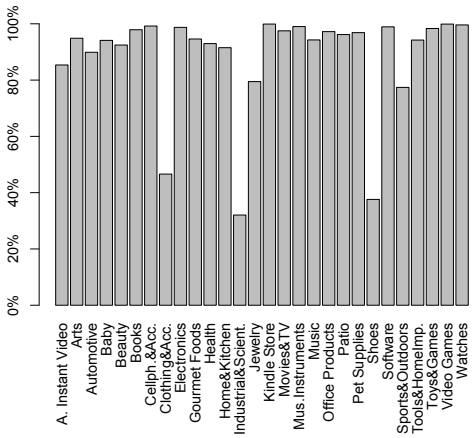
3.1 Duplicates The crawled datasets contain duplicates for items that are variants of the same product. This is possible due to, e.g., different size, color and material (for clothing), blue-ray and dvd versions (for movies), size (for shoes and gloves) etc. The users evaluate a specific variant of the product, e.g., "yellow pants of size small", but their reviews and ratings also appear in the Amazon webpages of all the different variants of the product. Since the data are acquired by crawling, all variants of the same product have exactly the same reviews and ratings, leading to duplicates. An example of duplicate records from the category "Industrial & Scientific" is shown in Table 2. User 1 has three entries with exactly the same timestamp (t_1), review text ("*As a professional plumber...*") and rating (3) for three different product IDs (items 1, 2, and 3). These products are variants in terms of size of the same product, "safety gloves".

3.2 Elimination of duplicates To clean the datasets, we kept only one of the different product variants and their corresponding reviews and ratings. There is no indication in the data about possible connections between different products. To identify the variants of a product, we had therefore to rely on the review text. Items with exactly the same reviews are *potentially* redundant, but not necessarily: there can be reviewers using the same text for different products. A closer inspection of the data showed that for some very generic and short reviews the text filter is not enough. To eliminate such cases we also used the review time, the numeric rating, and the id of the author. If the review text, author-id, rating, and time are identical, the items are deemed variants of the same product. Once we identify these sets of product variants, we keep only one variant of the product (the one with the smallest ID).

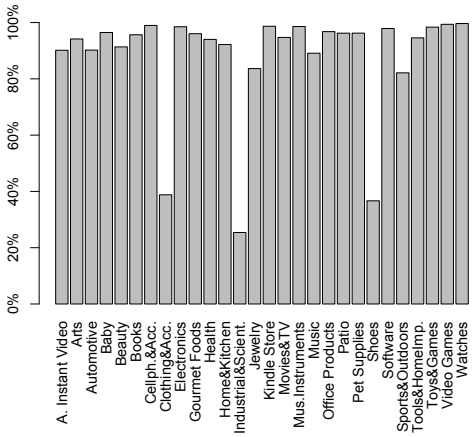
From the examples of Table 2, the reviews of user 1 have been identified as redundant, therefore only one instance (referring to item 1) is kept. As a counter

[1] `http://jmcauley.ucsd.edu/data/amazon/` — apparently they noticed the problem of duplicates and provide an updated version which was however not available for our experiments.

[2] McAuley and Leskovec [15] list a slightly different set of categories. For example, they mention a category "Furniture", which was not included in the datasets we could access. We can only speculate that the data of that category are distributed across other categories like "Home & Kitchen" or "Patio". Also, they do not mention the "Amazon Instant Video" category, which might have been part of their "Movies" category.



(a) % of retained items per category



(b) % of retained reviews per category

Figure 1: Effect of duplicate elimination on the datasets.

example, although user 3 has identical reviews for two products, the reviews have different timestamps and therefore they are not redundant.

3.3 Effects of duplicate elimination on the dataset Duplicate elimination results in less items as the different variants of an item, e.g., size- or color- or material-based, are eliminated. As a consequence, the amount of reviews is also reduced.

The effect on the amount of items is shown in Figure 1(a). All datasets are affected by the elimination, however to different degrees. The elimination has a huge impact on "Industrial & Scientific" where only the 32% of the initial items are retained. This implies that the removed 68% are just different variants of those items, in terms of, e.g., material, size, or color. The second most affected dataset is "Shoes" where duplicate elimination (e.g., different sizes of the same type of shoes) results in a much smaller dataset (roughly 37% of the initial items), followed by "Clothing Accessories" with 47% (e.g., due to different size and different colors). The datasets "Sports & Outdoor" and "Jewelry" follow with 77% and 79% retained items, respectively. Among the least affected datasets are "Watches" (99.6%), "Video Games" (99.9%), "Cell Phones and Accessories"(99.2%), and "Kindle Store" (99.9%).

The effect of the duplicate elimination on the number of reviews is shown in Figure 1(b). Again, the datasets are affected to different degrees but the most affected datasets in terms of items are most affected also in terms of reviews, namely, "Industrial & Scientific" (25%), "Shoes" (37%), "Clothing Accessories" (39%), "Sports & Outdoors" (82%), and "Jewelry" (84%). Among the least affected datasets are "Watches" (99.6%), "Video Games" (99.3%), "Cell Phones and Accessories"(98.9%), and "Kindle Store" (98.7%).

4 Impact of redundancy on the quality of recommendations

The random split of the original datasets (containing redundancies) into training, validation, and test sets incurs the possibility that these sets are effectively not disjoint. Having redundancies in the validation set strongly affects the selection of the best model, as the models are evaluated over instances seen during training. Redundancy therefore is manifested in our problem twice:

- in the model selection phase, as the training set upon which models are built and the validation set upon which the models are evaluated are not necessarily disjoint, and
- in the model testing phase, as the test set for which the performance of the best model is reported might be overlapping with the training set used for model construction and with the validation set used for model selection.

4.1 Experimental setup To evaluate the effect of redundancies on the quality of recommendations, we experiment with a variety of methods (as detailed in Section 2.2). In our experiments we used $k = 10$ factors for users and items, which achieved the best performance in previous evaluations [15]. For the *Offset*, *Latent Factors*, and *Hidden Factors* methods we used the authors' [15] implementations[3]. For the *Matrix Factorization* and *Biased Matrix Factorization* methods, we used the implementation in the MyMediaLite library [4].

[3]Available from: `http://cseweb.ucsd.edu/~jmcauley/code/code_RecSys13.tar.gz`

We evaluated the impact of data redundancy on all the datasets from the Amazon collection as listed in Table 1. As in the original paper [15], we *randomly* split each dataset into training, validation, and test sets. We use 80% of each dataset for training, up to a maximum of 2 million reviews. The remaining data are used for validation (10%) and test sets (10%). The validation set is used to select the model parameters: the model with the lowest error in the validation set is selected and its performance on the test set is reported in terms of the mean squared error (MSE). Different from the original evaluation setup [15], we repeated this process 10 times, following the 10-fold cross validation paradigm. At each run, we get an MSE score for the best model of the run. We report the *mean* and the *standard deviation* of the MSE over the 10 different runs.

The procedure was performed for both the original and the cleaned versions of the datasets. Since the split into training, validation, and test set was performed randomly, for the original datasets, variants of the same item can be present in the training, validation, and test sets, i.e., validation (and test) potentially use information that has been used already for training (or validation). For the cleaned datasets, the training, validation, and test sets are disjoint as redundancies have been eliminated as described in Section 3.2.

4.2 Effects on quality of recommendations We plot the mean and the standard deviation of the MSE over the 10 folds of the cross validation for the different datasets and the different recommendation methods, in the original datasets and after duplicate elimination, in Figure 2. We can see that the standard deviation is in most cases rather small (i.e., the performance over the 10 experiments in the cross validation splits was quite stable). Larger deviations are observed for the categories "Books" and "Music". A possible explanation is the inherent diversity of these categories with different genres, e.g., drama, romance, fiction etc. for books, or rock, pop, classic etc. for music. A difference of considerably more or considerably less correct predictions in some of the 10 repetitions can have a huge impact here, as MSE is rather sensitive to outliers. In most cases and for most methods, however, the performance is quite stable over the 10 repetitions.

For each dataset, we show (Figure 2) the performance on the original version and on the cleaned version. As we connect the means of the two variants of each dataset (with dotted lines), we can observe some intersections, which means that the ranking (best to worst) changed on the cleaned data vs. the original data, e.g., the "Clothing & Assessories" category case.

If the redundancy is high (i.e., many duplicate entries were removed from the original version), the performance gets typically worse (i.e., larger MSE) on the cleaned data. This is the case for datasets like "Industrial & Scientific", "Shoes", "Clothing Accessories" and "Sports & Outdoors". For the datasets with a relatively small amount of redundancy (i.e., the cleaned version does not differ too much from the original version, as there were not many duplicate entries to remove), the difference in performance is small. This category contains datasets like "Watches", "Video Games", "Cell Phones and Accessories" and "Kindle Store". In almost all cases, the field of the competitors is closer on the cleaned data, i.e., the performance differences are not as large as they appear to be on the original data.

Increase in MSE vs. data redundancy To get an overall picture of the redundancy versus MSE behavior, we plot in Figure 3 the relative increase of the error (MSE) between the original and the cleaned versions of each dataset (y-axis) versus the redundancy level (x-axis). The increase in the error is given as a ratio of the error on the cleaned dataset over the error on the original dataset:

$$\textit{relative increase}(MSE) \quad = \quad \frac{\overline{MSE}_{cleaned}}{\overline{MSE}_{orig}}$$

Values close to 1 along the y-axis therefore mean almost no change in the error before and after cleaning, while values around, say, 2 would mean there was a 100% increase of error (the error is twice as large on the cleaned dataset as on the original dataset). Along the x-axis, the datasets are arranged according to the relative amount (%) of retained reviews after cleaning. Smaller values correspond to less retained reviews and therefore to higher redundancy. Larger values indicate lower redundancy and, thus, a higher overlap between the original and the cleaned version. We can therefore easily interpret this plot as the x-axis points in the direction of lower redundancy in the original dataset, the y-axis points in the direction of degrading performance (increasing MSE).

We see a clear pattern for all methods: The performance degrades the stronger (higher y-axis values), the more redundancy was contained in the original dataset (lower x-axis values). For the Hidden Factors Model, the strongest degradation is 4, i.e., the error in the cleaned version is 4 times larger than the error in the original version (Industrial&Scientific), followed by a degradation of 3 (Shoes) and 2.8 (Clothing&Accessories). Different methods however differ in their sensitivity to the redundancy. For example, the method *Offset* shows almost no sensitivity while the method *Hidden Factors* is very sensitive to redundancy. The datasets with the largest amount of redundancy are "Cloth-

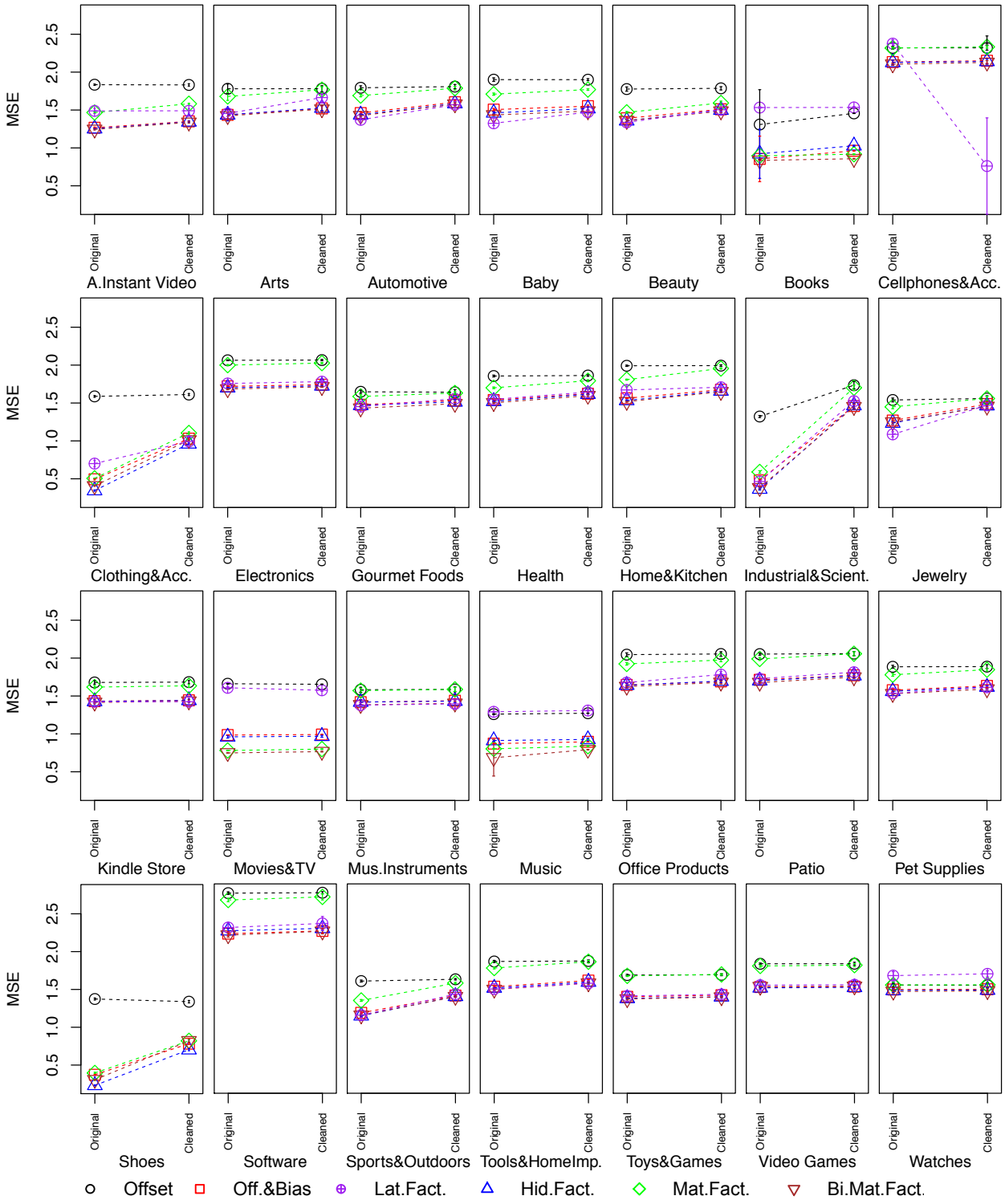


Figure 2: Comparison of all algorithms on datasets, original vs. cleaned.

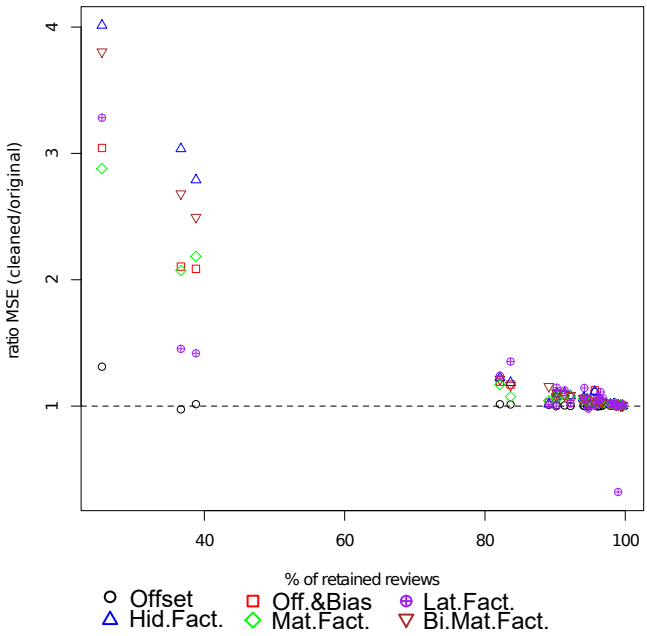


Figure 3: Relative increase of error (MSE) over the level of redundancy (percentage of retained reviews after cleaning) of the datasets.

ing&Accessories", "Industrial&Scientific", and "Shoes", followed by "Sports&Outdoors" and "Jewelry" (cf. Figure 1). In Figure 3, we find the first three between 20% and 40% of retained items, the next two slightly above 80%. On these five datasets, we can also observe the strongest changes in Figure 2. On "Clothing&Accessories", "Industrial&Scientific", and "Shoes", the degradation of the five best performing methods (*Offset* is always the worst, but relatively stable) is such that the error of the fifth ranked method on the original data is still better than the error of the best ranked method on the cleaned data which means that the results on the original data cannot establish a reliable ranking of the methods.

5 Discussion

As we saw in Figure 3, comparing the performance of methods on the cleaned vs. the original datasets, the performance of the methods tends to degrade more strongly with a higher level of redundancy (the smaller the amount of retained reviews, the higher the increase in MSE when comparing the cleaned vs. the original datasets performance).

A straightforward explanation would be to explain the strongest impact on the methods' performance on those datasets that were reduced most, simply by the smaller amount of training data after duplicate elimination. However, although all methods examined in this work are affected by the redundancies, the quantity to which the different methods are affected differs. Higher sensitivity to redundancy should therefore more sensibly be explained by a higher potential to overfit.

Typically, simpler, less adaptive methods are less prone to overfitting, whereas more complex methods are more sensitive to overfitting. While susceptibility to overfitting is not a desirable property for a learning method, it is not an extremely bad property either as long as one takes care of it by providing the correct training and test setup. A model should display a good performance on the training set but also on the (independent) test set. The later is an indication of the generalization power of the model, i.e., how well the model will perform on new instances. An overlap between training set and test set renders conclusions on the generalization ability of the model unreliable (and typically overly optimistic).

The degree of susceptibility to overfitting is different for different methods and can thus explain why we observe a quantitatively different behavior of the methods in the effect of duplicate elimination: the method *Offset* is most resistant to redundancy, while the method *Hidden Factors* is most sensitive.

The method *Offset* is a very simple method which relies solely on the numerical ratings and the prediction for an item is the average across all ratings in the training set. It would seem that removing redundant entries does not incur much change in the average score and therefore the *Offset* scores in the original versus cleaned datasets are very similar. The method *Offset with Bias* is only slightly more complex than the Offset but due to the bias component is rather more susceptible to high levels of redundancy. The method *Matrix Factorization* also works only with the numerical ratings. So does *Biased Matrix Factorization*, but this model additionally introduces compensation for rating bias. The method *Latent Factors* also works only with the numerical ratings but it takes also into account the user and item biases, and additionally the user-item similarity in the latent space. The method *Hidden Factors* considers both the numerical ratings and the textual reviews and therefore is affected a lot by redundancy. We can therefore rank the methods in the order as listed above according to increasing model complexity. We can observe an increasing susceptibility to performance degradation on the cleaned dataset versions from Offset (least susceptible) over Offset with Bias, Matrix Factorization, Biased Matrix Factorization, and Latent Factors, to Hidden Factors (most susceptible) in Figure 3. This order reflects the increasing complexity of the models.

In experimental evaluations of methods, a common

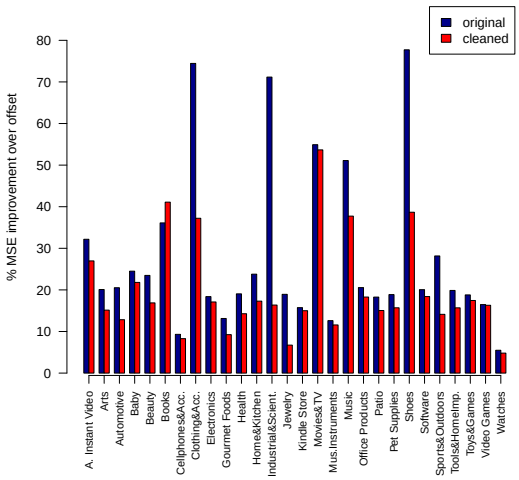


Figure 4: Relative improvement of the Biased Matrix Factorization method over the Offset method as baseline on the original vs. cleaned datasets. The general picture for the other methods would be the same.

way of presenting performance results of a new method is to compute the improvement over existing methods or over some baseline. In our experiments, the Offset method (as expected) performed worst on all original datasets and on almost all cleaned datasets. We therefore take Offset as baseline (as it is done in many publications) and plot the improvement of Biased Matrix Factorization as a competitor over Offset in Figure 4. (The plots for comparing the other methods' improvements over Offset look nearly identical and are omitted.) We can see that the improvements on the original datasets typically look much more impressive. The real improvements achieved by recent papers that used the Amazon reviews datasets for evaluation might be much more moderate.

6 Conclusions

6.1 Summary We investigated the impact of redundancies in a popular dataset collection on the evaluation of recommendation models. We observe that conclusions on performance comparisons between different models, stated in the literature, may change when tested on cleaned data. This effect is apparently (and intuitively understandably) stronger for more complex methods. This observation could hint a stronger susceptibility of these methods to overfitting.

Our results suggest that these datasets should be used only after duplicate elimination, and that the improvements reported for recent methods, as tested on the original Amazon reviews datasets, are to be taken with a grain of salt. Methods that showed strong improvements on these datasets may simply be more susceptible to overfitting and their improvements may have been assessed overly optimistically due to the redundancies in the datasets. Our results do not imply, however, that these recent methods do not constitute methodological improvements. In any case, the assessment of data mining results should be conducted more carefully, by considering both the quality of the data and the evaluation process.

6.2 So what? The implications of our observations are not restricted to the obvious conclusion that datasets in general should be carefully preprocessed, and redundancies should be removed or get otherwise special treatment when splitting a dataset into (supposedly) disjoint subsets, and that the Amazon reviews datasets need special care in particular when they will be used in future experiments. We should also inspect evaluation results in the literature in general with much more diligence. Based on our experiments, we can observe a change of conclusions (at least quantitatively, in some cases also qualitatively) compared to previously reported results.

Questions arise naturally as to possibilities to automatically control datasets for potential problems. Alas, this is not easily possible. Duplicates are just one possible problem and we do not know which other problems might be hidden. If we focus on the problem of duplicates, we have to admit that our approach to duplicate elimination is quite heuristic. One could follow more or less radical strategies of removing duplicates. One of the anonymous reviewers suggested that we could, instead of removing duplicates, specialize the splitting procedure of the cross-validation such that the time stamps are taken into consideration: earlier instances should be in the training set and later instances in the test set. This would avoid using the duplicated entries in both, training and test set. Such an approach would also have the nice property of being realistic in the sense that the algorithms indeed shall predict the future. However, if all data instances should be used in the same proportion, only one such split is possible. Another problem of duplicates would remain: they implicitly lead to higher weights of the (different) duplicated examples during training and during testing. In the Amazon datasets, this should not be tolerated, as the duplicates do not constitute more important examples, just examples that come in more variations (such as color or size).

Going beyond the results of our case study focused on the Amazon dataset collection, let us remark with a more general perspective that the quality of evaluations and the reliable assessment of data mining methods is an extremely important problem nowadays as data

mining is more and more mainstream and users of methods are not necessarily data mining experts. To make things even more complicated, the explosion of online networks has lead to the creation of self-crawled datasets. As we demonstrated, this can lead to problems as what one gets through the browser interface does not necessarily reflect the semantics of the data. In the Amazon case we presented here, the problem likely occurred due to Amazon showing the same reviews for variants of the same product. Similar problems can be easily manifested in other applications too. For example, Twitter, one of the most popular data sources for data mining and social media research, comprises a characteristic example of content redundancy as a lot of its users re-tweet tweets from other users. In some of the studies, e.g., for sentiment classification [6] or spam detection [19], such redundancies are removed. However, redundancy as such is not necessarily a bad thing, depending on the use case. For example, in Twitter the number of re-tweets comprises a measure of popularity for a tweet. So, duplicates can be genuine parts of a dataset and removing them would change the inherent dataset characteristics.

Let us emphasize that the service of McAuley and Lescovec, providing their benchmark data publicly, constitutes an important and valuable service to the community. In other areas such as IR, NLP, or computer vision, the practice of commonly agreed upon benchmark data is much more established. The area of data mining will certainly benefit from having established benchmark data for various tasks.

The community should always remain careful in using data for evaluation and competitive comparison of methods and appreciate independent evaluation studies, though. Evaluation is from a scientific point of view not less but more important than the design of ever newer and "better" (really?) methods, and is typically challenging and far from trivial, as has been discussed for other areas of data mining as well [1, 2, 7, 9, 20, 22].

References

[1] A. Bifet, J. Read, I. Zliobaite, B. Pfahringer, and G. Holmes. Pitfalls in benchmarking data stream classification and how to avoid them. In *Proc. ECML PKDD*, pages 465–479, 2013.

[2] G. O. Campos, A. Zimek, J. Sander, R. J. G. B. Campello, B. Micenková, E. Schubert, I. Assent, and M. E. Houle. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Min. Knowl. Disc.*, 2016.

[3] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan. Collaborative filtering recommender systems. *Human–Computer Interaction*, 4(2):81–173, 2010.

[4] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. MyMediaLite: a free recommender system library. In *Proc. RecSys*, 2011.

[5] R. Gemulla, E. Nijkamp, P. J. Haas, and Y. Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In *Proc. SIGKDD*, 2011.

[6] A. Go, R. Bhayani, and L. Huang. Twitter sentiment classification using distant supervision. *Processing*, pages 1–6, 2009.

[7] F. Janssen and J. Fürnkranz. A re-evaluation of the over-searching phenomenon in inductive rule learning. In *Proc. SDM*, pages 329–340, 2009.

[8] R. Johnson and T. Zhang. Semi-supervised convolutional neural networks for text categorization via region embedding. In *Proc. NIPS*, pages 919–927, 2015.

[9] H.-P. Kriegel, E. Schubert, and A. Zimek. The (black) art of runtime evaluation: Are we comparing algorithms or implementations? *Knowl. Inf. Syst.*

[10] D. Lim, J. McAuley, and G. Lanckriet. Top-n recommendation with missing implicit feedback. In *Proc. RecSys*, pages 309–312, 2015.

[11] J. McAuley, J. Leskovec, and D. Jurafsky. Learning attitudes and attributes from multi-aspect reviews. In *Proc. ICDM*, pages 1020–1025, 2012.

[12] J. McAuley, R. Pandey, and J. Leskovec. Inferring networks of substitutable and complementary products. In *Proc. SIGKDD*, pages 785–794, 2015.

[13] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. Image-based recommendations on styles and substitutes. In *Proc. SIGIR*, pages 43–52, 2015.

[14] J. J. McAuley and J. Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proc. WWW*, 2013.

[15] J. J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proc. RecSys*, pages 165–172, 2013.

[16] M. Pourrajabi, D. Moulavi, R. J. G. B. Campello, A. Zimek, J. Sander, and R. Goebel. Model selection for semi-supervised clustering. In *Proc. EDBT*, 2014.

[17] S. Rendle and L. Schmidt-Thieme. Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In *Proc. RecSys*, pages 251–258, 2008.

[18] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors. *Recommender Systems Handbook*. Springer, 2011.

[19] S. Sedhai and A. Sun. Hspam14: A collection of 14 million tweets for hashtag-oriented spam research. In *Proc. SIGIR*, pages 223–232, 2015.

[20] L. Swersky, H. O. Marques, J. Sander, R. J. G. B. Campello, and A. Zimek. On the evaluation of outlier detection and one-class classification methods. In *Proc. DSAA*, pages 1–10, 2016.

[21] Y. Yang, Y. Yan, M. Qiu, and F. S. Bao. Semantic analysis and helpfulness prediction of text for online product reviews. In *Proc. ACL*, pages 38–44, 2015.

[22] A. Zimmermann. The data problem in data mining. *SIGKDD Explor.*, 16(2):38–45, 2014.