# PERSONALIZED RANKING OF MOVIES: EVALUATING DIFFERENT METADATA TYPES AND RECOMMENDATION STRATEGIES

**Frederico Araujo Durão**
Universidade Federal da Bahia/Brazil
freddurao@dcc.ufba.br

**Renato Dompieri Beltrão**
Universidade de São Paulo/Brazil
rdompieri@grad.icmc.usp.br

**Bruno Souza Cabral**
Universidade Federal da Bahia/Brazil
brunocabral@dcc.ufba.br

**Marcelo Garcia Manzato**
Universidade de São Paulo/Brazil
mmanzato@icmc.usp.br

***Abstract*: This paper proposes a study and comparison among a variety of metadata types in order to identify the most relevant pieces of information in order to identify the most relevant pieces of information in personalized ranking of movie items. We used four algorithms available in the literature to analyze the descriptions, and compared each other using the metadata extracted from two datasets, namely MovieLens and IMDB. As a result of our evaluation, we found out that the movies' genres and actors are the kind of description that generates better predictions for the considered content-based recommenders.**

***Keywords:* Recommender systems; Metadata; Matrix factorization; Latent factors.**

## I. INTRODUCTION

Due to the large amount of information present in the World Wide Web, we observe a difficulty for users to deal with this huge quantity of content available. This problem is known as information overload, and a tool that helps individuals to manage such content is recommender systems. There are a number of ways to build recommender systems; basically they are classified as content-based filtering, collaborative filtering and the combination of both of them [1], [2].

Content-based filtering recommends multimedia content to the user based on a profile containing information regarding the content, such as genre, keywords, subject, etc. These metadata are weighted according to past ratings, in order to characterize the user's main interests. A problem with this approach is over- specialization, which happens when the system recommends only items that are too similar to the items already rated [1]. Another issue is the limited metadata about the content, since the interest profile is obtained through these descriptions. In case the item description is poor, it will barely be considered for recommendation.

An alternative to this problem is the collaborative filtering, which is based on clusters of users or items. In the first case, items that are appreciated by a group of users with the same interests are recommended to a particular user of that group. In the second case, if two items have the same evaluation by different users, then these items are considered similar, so it is expected that the users have likely tastes for similar items [2].

One disadvantage of collaborative filtering is the computational effort spent to calculate similarity between users and/or a feature space containing topics of interest [5], [6], [10], [12]. Nevertheless, other challenges have to be dealt with, such as sparsity, over fitting and data distortion caused by imputation methods [5].

Considering the limitations and challenges depicted above, hybrid recommenders play an important role because they group together the benefits of content based and collaborative filtering. It is known that limitations of both approaches, such as the cold start problem, overspecialization and limited con- tent analysis, can be reduced when combining both strategies into a unified model [1]. However, most recent systems which exploit latent factor models do not consider the metadata associated to the content, which could provide significant and meaningful information about the user's interests.

In related work [1], [3], [9], [7], we verify a set of recommender algorithms which exploit latent factors, collaborative filtering, metadata awareness and implicit feedback. However, there is a lack of study about which metadata type generates the best results in the domain of movies. In this way, the present paper aims to compare a variety of movie metadata with four recommendation algorithms in order to identify those pieces of information that are more important in the process of recommending movies to the user.

This work is structured as follows: in Section II we describe the models considered in this evaluation; in Section III we depict how the metadata is extracted; Section IV presents the evaluation of different metadata applied to the four considered algorithms; and finally, in Sections V and V we discuss the final remarks, future work and acknowledgements.

## II. CONSIDERED MODELS

In this section we describe in more details the models used to study and compare the different types of metadata considered in this paper.

### A. *Notation*

The Following the same notation in [5], [8], we use special indexing letters to distinguish users, items and attributes: a user is indicated as $u$, an item is referred as $i, j, k$ and an item's attribute as $g$. The notation $r_{ui}$ is used to refer to explicit or implicit feedback from a user $u$ to an item $i$. In the first case, it is an integer provided by the user indicating how much he liked the content; in the second, it is just a boolean indicating whether the user consumed or visited the content or not. The prediction of the system about the preference of user $u$ to item $i$ is represented by $\hat{r}_{ui}$, which is a floating point value r calculated by the recommender algorithm. The set of pairs (u, i) for which $r_{ui}$ is known is represented by the set $K = \{(u, i) \mid r_{ui} \text{ is known}\}$.

Additional sets used in this paper are: N (u) to indicate the set of items for which user u provided an implicit feedback, and $\bar{N}$(u) to indicate the set of items that is unknown to user *u*.

### B. *BPR-Linear*

The BPR-Linear [3] is an extension of matrix factorization optimized for Bayesian Personalized Ranking (BPR-MF) [11] that can deal with the cold-start problem, yielding accurate and fast attribute-aware item recommendation methods based on a linear mapping for score estimation.

Bayesian Personalized Ranking (BPR) is a framework for optimizing different kinds of models based on training data containing implicit feedback or other kinds of implicit and explicit (partial) ranking information. It was proposed by Rendle et al. [11] to address the issue that happens when training an item recommendation model using implicit feedback based only on positive/negative data. The model will be fitted to provide positive scores to the observed items, while considering items not observed as negative. However, such assumption is inaccurate because a not observed item may be due to the fact it was unknown to the user.

Considering this problem, instead of training the model using only the user-item pairs, Rendle et al. proposed also to consider the relative order between a pair of items, according to the user's preferences. It is inferred that if an item $i$ has been viewed by user $u$ and $j$ has not ($i \in N(u)$ *and* $j \in \bar{N}(u)$), then $i >_u j$, which means that he prefers $i$ over $j$. Figure 1 presents an example of this method.

The key idea is to consider entity pairs instead of single entities in its loss function, allowing the interpretation of positive- only data as partial ranking data. The user item preference estimation, is based on a Bayesian analysis using the likelihood function for $p(i >_u j \mid \theta)$ and the prior probability for the model parameter $p(\theta)$. The final optimization criterion, BPR-Opt, is defined as:

$$\text{BPR-Opt} := \sum_{(u,i,j) \in D_K} \ln \sigma(\hat{s}_{uij}) - \Lambda_\Theta \|\Theta\|^2 \quad , \qquad (1)$$

where $\hat{s}_{uij} := \hat{r}_{ui} - \hat{r}_{uj}$ and $D_k := \{(u,i,j) \mid i \in N(u) \;\&\; j \in \bar{N}(u)\}$. The symbol $\Theta$ represents the parameters of the model,$\Lambda_\Theta$ is a regularization constant, and $\sigma$ is the logistic function, defined as: $\sigma(x) = \frac{1}{(1+e^{-x})}$.

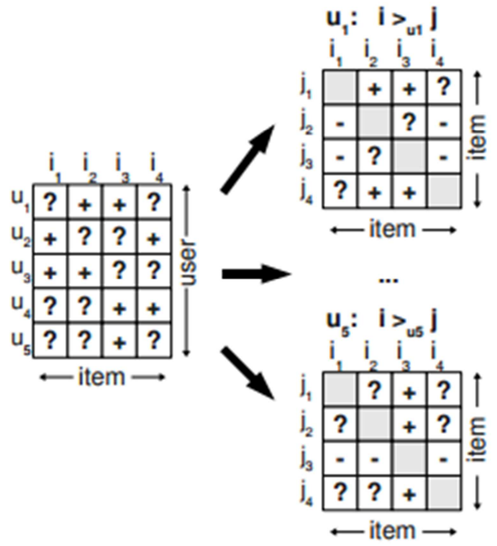


Fig. 1. Extracted from Rendle et al., the left-hand side table represents the observed data *K*. On right-hand side, after applying a user-specific pairwise relation i > *u* j, the plus signal indicates that user *u* has more interest in item *i* than *j*; the minus signal indicates he prefers item *j* over *i*; and the interrogation mark indicates that no conclusion can be inferred between the items.

For learning the model, the authors use a variation of the stochastic gradient descent technique, denominated LearnBPR, which randomly samples from $D_K$ to adjust $\Theta$. Algorithm 1 shows an overview of the algorithm, where $\alpha$ is the learning rate.

**Input**: $D_K$
**Output**: Learned parameters $\Theta$
Initialize $\Theta$ with random values
**for** *count* = *1,...,#Iter* **do**
  draw $(u, i, j)$ from $D_K$
  $\hat{s}_{uij} \leftarrow \hat{r}_{ui} - \hat{r}_{uj}$
  $\Theta \leftarrow \Theta + \alpha \left( \frac{e^{-\hat{s}_{uij}}}{1+e^{-\hat{s}_{uij}}} \cdot \frac{\partial}{\partial\Theta} \hat{s}_{uij} - \Lambda_\Theta($
**end**

**Algorithm 1:** Learning through Learn

Gantner et al. [3] address the case where new users and items are added by computing first the latent feature vectors from attributes like the user's age or movie's genres, and then using those estimated latent feature vectors to compute the rating from the underlying matrix factorization (MF) model.

The score estimation using the item attributes is obtained by:

$$\hat{r}_{ui} = \phi_f(\vec{a}_i) = \sum_{g=1}^{n} w_{ug} a_{ig} \quad ,$$

Where $\emptyset_f: \mathbb{R}^n \rightarrow \mathbb{R}$ is a function that maps the item attributes to the general preferences $\hat{r}_{ui}$ and $\underset{a_i}{\rightarrow}$ is a boolean vector of size *n* whose each element $a_{ig}$ represents the occurrence or not of an attribute, and $w_{ug}$ is a weight matrix learned using LearnBPR.

### C. BPR-Mapping

The BPR-Mapping [3] was also proposed by Gantner et al.; the key difference is that it uses a different attribute-to-features mapping procedure. Gantner et al. explained that one way to learn suitable parameters for the linear mapping functions is optimizing the model for the (regularized) squared error on the latent features, and a ridge regression was used. In addition, a stochastic gradient descent was used for training because of the enormous number of input variables. Nevertheless, this approach leads to a sub-optimal performance. Thereafter, a linear mapping optimized for BPT-Opt was proposed and is what is used in BPR-Mapping.

### D. MABPR

One disadvantage of the previous BPR algorithms is that they are not able to infer any conclusion when the items *i* and *j* are known (or both are unknown).In other words, if an item has been viewed by the user, it is possible to conclude that this content is preferred over all other unknown items, as it aroused a particular interest to him than the others. On the other hand, when both items are known (or both are unknown), it is not possible to infer which one is preferred over the other because the system only has the positive/negative feedback from the user. Consequently, those pairs which belong to the same class (positive or negative) will not be able to be ranked accordingly, as the model will be learned only by using the specific case where one item is known and the other is not.

To overcome this limitation, Manzato et al. (manuscript in preparation) proposed an extension to the BPR technique which also considers metadata from items in order to infer the relative importance of two items.

It starts by redefining the set DK which contains the data used during training to $D'_k := \{(u,i,j) | \ i \in N(u) \ \& \ j \in \bar{N}(u) \ \boldsymbol{or} \ i \in N(u) \& \ j \in N(u) \cup N(u) \& \ |G(i)| > 0 \ \& \ |G(j)| > 0 \}$ to consider the metadata available in the specified case, while also considering items without descriptions.

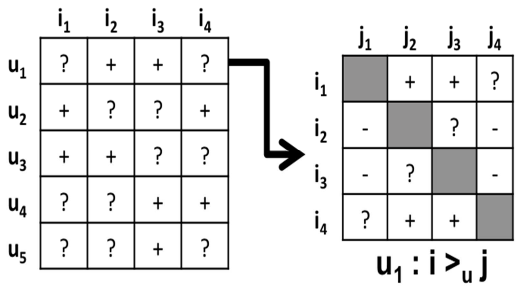


Fig. 2. As an extension to Rendle et al. approach, we also consider the metadata describing items *i and j* when both are known ($i \in N(u) \& \ j \in N(u)$). The function $\delta(i,j)$ returns positive whether user *u* prefers the description of item *i* over the description of item *j*, and negative otherwise.

Figure 2 shows how the proposed extension affects the relationship between items *i* and *j* with respect to the preferences of user *u*. Because items $i_2$, $i_4$ and $i_5$ are known, the system has to analyze their metadata to infer which one is preferred over the other. This is the role of function $\delta(i,j)$,, which is defined as:

$$\delta(i,j) = \begin{cases} + & \text{if} \quad \varphi(u,i) > \varphi(u,j), \\ - & \text{if} \quad \varphi(u,i) < \varphi(u,j), \\ ? & \text{otherwise,} \end{cases} \qquad (3)$$

where $\varphi(u,.)$ is defined as:

$$\varphi(u,.) = \frac{1}{|G(.)|} \sum_{g \in G(.)} w_{ug} \quad , \qquad (4)$$

and $w_{ug}$ is a weight indicating how much *u* likes a description $g \in G(.)$.

This approach enhances the BPR algorithm with further insight about the user's preferences by considering his personal opinions about particular descriptions of items. Such metadata can be of any type: genres of movies/music, keywords, list of actors, authors, etc.

The mechanism used to infer such opinions $w_{ug}$ by analyzing only the training data is accomplished by adopting a linear attribute-to-feature mapping similar to the one proposed by Gantner et al. [3], and then, optimizing the parameters using the LearnBPR algorithm. It is used the score estimation equation 2, and in order to learn $w_{ug}$ using LearnBPR, is computed the relative importance between two items:

$$\begin{aligned} \hat{s}_{uij} &= \hat{r}_{ui} - \hat{r}_{uj} \\ &= \sum_{g=1}^{n} w_{ug} a_{ig} - \sum_{g=1}^{n} w_{ug} a_{jg} \\ &= \sum_{g=1}^{n} w_{ug} (a_{ig} - a_{jg}) \quad . \end{aligned}$$

Finally, the partial derivative with respect to $w_{ug}$ is taken:

$$\frac{\partial}{\partial w_{ug}} \hat{s}_{uij} = (a_{ig} - a_{jg}) \quad , \tag{6}$$

which is applied to Algorithm 1 considering that $\theta = (w_*)$ for all set of users and descriptions.

## E. *MostPopularByAttributes*

This is a simple algorithm similar to the "Same artist - greatest hits" baseline presented on McFee et al. [9]. It recommends a ranked item list ordered by popularity, considering attributes that the user had seen previously, followed by the remaining items also ordered by popularity. For instance, if a user has listened only to Rock music, it will recommend first the most popular Rock songs, followed by other genres.

## III. METADA EXTRACTION

For the tests, we used the 100k MovieLens database combined with Internet Movie Database (IMDB) in order to infer which is the best algorithm in movie recommendations. Once the MovieLens dataset has little information about the movies, we then extracted additional information from IMDB database, thus enriching the movie dataset information. Figures 3 and 4 illustrate the items present in each dataset.

| **TITLE** | **MOV_KEYW** | **AKA_TITLE** | **COMPANY_NAME** | **MOV_CIA** |
|---|---|---|---|---|
| id | id | id | id | id |
| title | movie_id | movie_id | name | movie_id |
| imdb_index | keyword_id | title | country_code | company_id |
| kind_id | **AKA_NAME** | imdb_index | imdb_id | company_type_id |
| production_year | id | kind_id | name_pcode_nf | note |
| imdb_id | person_id | production_year | name_pcode_sf | **MOVIE_INFO** |
| phonetic_code | name | phonetic_code | **CAST_INFO** | id |
| episode_of_id | imdb_index | episode_of_id | id | movie_id |
| season_nr | name_pcode_cf | season_nr | person_id | info_type_id |
| episode_nr | name_pcode_nf | episode_nr | movie_id | info |
| series_years | surname_pcode | **COMP_CAST** | person_role_id | note |
| **CHAR_NAME** | **KEYWORD** | id | note | **LINK_TYPE** |
| id | id | movie_id | nr_order | id |
| name | keyword | subject_id | role_id | link |
| imdb_index | phonetic_code | status_id | **MOVIE_LINK** | **ROLE_TYPE** |
| imdb_id | **INFO_TYPE** | **PERSON_INFO** | id | id |
| name_pcode_nf | id | id | movie_id | role |
| surname_pcode | info | person_id | linked_movie_id | **CIA_TYPE** |
| **NAME** | **MOV_INFO_IDX** | info_type_id | link_type_id | id |
| id | id | info | **CCAST_TYPE** | kind |
| name | movie_id | note | id | |
| imdb_index | info_type_id | **KIND_TYPE** | kind | |
| imdb_id | info | id | | |
| name_pcode_cf | note | kind | | |
| name_pcode_nf | | | | |

Fig. 3. The IMDB database.

| **MOVIE** | | | | |
|---|---|---|---|---|
| id | Action | Crime | Horror | Thriller |
| title | Adventure | Documentary | Musical | War |
| production_year | Animation | Drama | Mystery | Western |
| imdb_url | Children | Fantasy | Romance | |
| Unknow | Comedy | Noir | SciFi | |
| **RATING** | | | | |
| userid | movieid | rating | timestamp | |
| **REL_MLENS_IMDB** | | | | |
| mlens_id | imdb_id | | | |

Fig. 4. The MovieLens database.

The most relevant data contained in these sets are the indexes because through them we can align the information in both datasets. Since the indexes of IMDB and Movielens are not the same, their titles and years present in MovieLens are used to identify the movies index in IMDB and recover the information we wanted. It was necessary to manipulate the data in MovieLens because the movie titles were written in English form (e.g. Godfather, The). So, we fixed these names to the form used in IMDB (e.g. The Godfather). The discovery of these indexes enabled us to extract the information we needed, i.e. genre, actor, writer, director and keyword. With this metadata we created tables of indexes, connecting the movies with their metadata. As we only used the movies from MovieLens dataset, the additional information extracted from IMDB was incorporated to the MovieLens dataset.

## IV. EVALUATION

In the evaluation presented in this paper, we compared five different types of metadata: actors, directors, genres, keywords and writers using the recommendation algorithms previously described in Section II. These algorithms were implemented using MyMediaLite library [4], which provides various options to matrix factorization and error measure. To measure the accuracy of recommendations, we used the Mean Average Precision (MAP).

The tests were executed with our improved database of MovieLens 100k, which contains 100,000 ratings of 943 users on 1682 movies. Each user rated at least 20 movies freeing us from the cold start problem. Worth mentioning that only three movies did not have additional information extracted from IMBD, which did not impact the results.

After executing the algorithms for each metadata and with different numbers of latent factors in the range [10..100], we compared the best values returned by MAP in each algorithm and each metadata. The goal was to infer the most suitable in each case. The obtained results are illustrated in the Figure 5.

TABLE I. MAP VALUES

| | Actor | Director | Genre | Keyword | Writer |
|---|---|---|---|---|---|
| MABPR | 0.253 | 0.252 | 0.254 | 0.251 | 0.251 |
| BPR-Linear | 0.042 | 0.039 | 0.039 | 0.055 | 0.044 |
| BPR-Mapping | 0.255 | 0.254 | 0.250 | 0.251 | 0.251 |
| MostPopular | 0.041 | 0.061 | 0.034 | 0.021 | 0.021 |

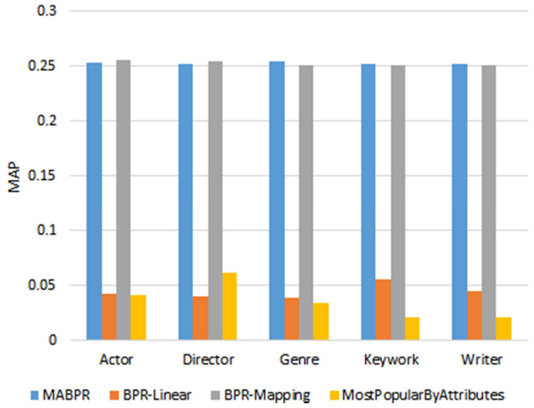


Fig. 5. Comparison among algorithms.

The algorithms MABPR and BPR-Mapping returned better results according to the MAP measure. These two algorithms generated a MAP value greater than 0.250 in all tested cases, while the others reached a maximum of 0.06. In particular, the best results were achieved when the BPR-Mapping algorithm was combined with the actor metadata, or when the MABPR algorithm was combined with the genre metadata. These combinations returned MAP values of 0.2552 and 0.2531 respectively.

Regarding the analyzed metadata, none of the algorithms returned the best recommendation for all tested cases. As shown, the results are balanced and there is a variation of the best metadata in each algorithm. However, this occurs because each method has its own purposes. For example, the *MostPopularByAttributes* was originally proposed for recommending popular songs from an artist that the user already liked [9]. Thus, we expect that the entities directors and actors to produce a better result over other metadata types in this algorithm.

When analyzing the results, it is possible to conclude that the metadata with the best recommendations for one algorithm is not equivalent in other algorithm. This behavior is observed by analyzing the MAP values among the tested algorithms. An example is the fact that keyword is the metadata which returned the highest MAP in the algorithm BPR-Linear with MAP 0.05054, and the genre is the metadata which returned the highest MAP value in the algorithm MABPR with MAP 0.25314. Thus, it is possible to note that some algorithms work better when using more general descriptions (e.g. genres), whereas other produce better results when using more specific descriptions (e.g. keywords). Nevertheless, although different metadata vary differently in each analyzed algorithm, it is clear that the genre metadata has a bigger relevance than the single keyword, because according to the MAP measure, it returns better recommendations, as it describes the whole content in general, and not a single subject of the movie. Thus, instead of searching a metadata that prevails over all algorithms, we searched for better recommendations. Finally, we conclude that best recommendations are achieved when the algorithm BPR- Mapping uses the actor's metadata and when the algorithm MABPR uses genres metadata.

## V. FINAL REMARKS

This paper shows four different algorithms that use movie metadata to generate recommendations of movies. One of these algorithms consists of an extension we made on the BPR technique, in order to consider metadata when two items are known by the user. These algorithms are combined with five types of metadata in order to infer which achieves better results according to MAP measure. After comparing the metadata with four different algorithms, we can conclude that the best algorithms in our tests are BPRMFAttr and BPRMF Mapping, all metadata achieves the best results with them. Also, using actor metadata in BPRMF Mapping algorithm, it produces better recommendations than other types of metadata, and genre produces the best recommendations when using MABPR algorithm.

As future work, we plan to evaluate the algorithms with a combination of two or more types of metadata in order to verify whether multimodal information can generate better recommendations. In order to do so, it will be necessary to extend the algorithms to exploit the descriptions in an effective fashion.


## Acknowledgment

The author would like to thank the financial support from CNPq, project number 1169.


## References


[1] G. Adomavicius and A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extension s. IEEE Transactions on Knowledge and Data Engineering, 17(6):734-749, 2005.

[2] M. D. Ekstrand, J. Riedl, and J. A. Konstan. Collaborative filtering recommender systems. Foundations and Trends in Human-Computer Interaction, 4(2):175-243, 2011.

[3] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recom-mendations. In 2010 IEEE 10th International Conference on Data Mining (ICDM), pages 176-185, dec. 2010.

[4] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. MyMediaLite: A free recommender system library. In Proceedings of the 5th ACM Conference on Recommender Systems, RecSys '11, pages 305-308, New York, NY, USA, 2011.

[5] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. ACM Trans. Knowl. Discov. Data, 4(1):1:1-1:24, Jan. 2010.

[6] M. Kurucz, A. A. Bencz´ , and B. Torma. Methods for large scale svd ur with missing values. In KDD Cup Workshop 2007, 2007.

[7] M. G. Manzato. Discovering Latent Factors from Movies Genres for Enhanced Recommendation. In Proceedings of the 6th ACM Conferece on Recommender Systems, RecSys '12, pages 249-252, New York, NY, USA, 2012.

[8] M. G. Manzato. gSVD++: supporting implicit feedback on recom- mender systems with metadata awareness. In Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13, pages 908- 913, New York, NY, USA, 2013. ACM.

[9] B. McFee, T. Bertin-Mahieux, D. P. Ellis, and G. R. Lanckriet. The million song dataset challenge. Proceedings of the 21st international conference companion on World Wide Web - WWW '12 Companion, page 909, 2012.

[10] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In Proc. KDD Cup Workshop at SIGKDD'07, 13th ACM Int. Conf. on Knowledge Discovery and Data Mining, pages 39-42, 2007.

[11] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI '09, pages 452-461, Arlington, Virginia, United States, 2009. AUAI Press.

[12] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of Dimensionality Reduction in Recommender System - A Case Study. In Proceedings of ACM SIGKDD Conference on Knowledge Discovery in Databases, Boston, MA, USA, 2000.