Recommending for Disloyal Customers with Low Consumption Rate

Ladislav Peska and Peter Vojtas

Faculty of Mathematics and Physics
Charles University in Prague
Malostranske namesti 25, Prague, Czech Republic
{Peska,vojtas}@ksi.mff.cuni.cz

Abstract. In this paper, we focus on small or medium-sized e-commerce portals. Due to high competition, users of these portals are not too loyal and e.g. refuse to register or provide any/enough explicit feedback. Furthermore, products such as tours, cars or furniture have very low average consumption rate preventing us from tracking unregistered user between two consecutive purchases. Recommending on such domains proves to be very challenging, yet interesting research task. For this task, we propose a model coupling various implicit feedbacks and object attributes in matrix factorization. We report on promising results of our initial off-line experiments on travel agency dataset. Our experiments corroborate benefits of using object attributes; however we are yet to decide about usefulness of some implicit feedback data.

Keywords: Recommender systems, implicit feedback, content-based attributes, e-commerce, matrix factorization.

1 Introduction

Recommending on the web is both an important commercial application and popular research topic. The amount of data on the web grows continuously and it is nearly impossible to process it directly by a human. The keyword search engines were adopted to fight information overload but despite their undoubted successes, they have certain limitations. Recommender systems can complement onsite search engines especially when user does not know exactly what he/she wants.

1.1 Our Motivation

Recently, a lot of attention was attracted by the NetFlix prize[1] aiming to predict future user rating based on previously rated objects. Our scenario is however substantially different.

First, we want to recommend on e-commerce applications, where users are generally less willing to provide explicit feedback (besides they are usually not

[1] `http://www.netflixprize.com/`

capable to provide relevant feedback for products, they did not buy or test yet). The vast majority of e-shops do not force users to register at all, which makes it difficult to track them. Combination of unregistered user and low consumption rate on particular product domains (tours, cars, furniture, specialized sport goods etc.) prevents us in many cases from effectively tracking consecutive purchases of the user.[2]

Given the described preconditions, the vast majority of users appear to be new users exacerbating the cold start problem. We cannot hope for tens of ratings as in multimedia portals, but rather need to cope with a few visited pages.

On the other hand, we can monitor additional data to improve our prospects. It is possible to record various types of implicit feedback (page-views, time on page, mouse usage, scrolling etc.) or track user behavior on category pages. Objects of the e-shops also contain vast number of content-based attributes..

1.2 Main Contribution

The main contributions of this paper are:

- Identifying challenging domain for recommender systems.
- Proposed recommending methods based on matrix factorization incorporated with implicit feedback and object attributes.
- Off-line experiments on travel agency dataset.
- Travel agency dataset for further experiments.

The rest of the paper is organized as follows: review of some related work is in section 2. In section 3 we describe travel agency dataset and in section 4 methods how to incorporate it into matrix factorization. Section 5 contains results of our off-line experiment on a travel agency dataset. Finally section 6 concludes our paper and points to our future work.

2 Related Work

The area of recommender systems has been extensively studied recently and it is out of scope of this paper to provide more elaborated overview. We suggest Konstan and Riedl [6] paper as a good starting point.

Implicit user feedback did not draw too much attention during early research on recommender systems, but grows on importance recently mainly in commercial area. A well known attempt to categorize various types of implicit feedback was proposed by Kelly and Teevan [5]. The list of feedback types presented in this study is due to the age of the survey not exhaustive any more (e.g. commencement of social networks and related activities), however it still presents a good starting point.

[2] Unregistered users are usually tracked by cookies stored within the browser. The cookie however may be lost for various reasons e.g. flushing data stored in browser, switching browsers or updating hardware. For a fixed user, the average distance between two consecutive purchases of e.g. a tour is approximately one year.

An approach to categorize methods for collecting implicit feedback was made by Gauch et al. [3]. Authors however didn't mention one important method: tracking user directly by a subprogram, e.g. a JavaScript code, deployed on the current website. Such approach needs cooperation from the site owners and it is possible to track only within cooperating sites, but data are received about vast majority of the users and it is possible to monitor more different information than e.g. by analyzing weblog files. Such user-tracking is also well suited to form a knowledge base for site-wide recommender systems.

Although there is quite large variety in implicit feedback types, publicly available datasets e.g. Last.fm[3] usually provides only very limited (if any) implicit feedback. As result, many research papers consider only binary implicit feedback and only one or a few types of feedback [4], [10].

Our feedback collecting method is based on JavaScript tracking of multiple feedback types on a single site. We have tested usefulness of various implicit feedback types in an on-line experiment in our previous work [11] and described method for learning user preferences from implicit feedback in [12].

Matrix factorization techniques [8] are currently main-stream algorithm to learn user preferences gaining their popularity during NetFlix prize. We use content-boosted matrix factorization as proposed in Forbes and Zhu [2] as one of the preference learning methods. Content-based and hybrid recommender systems may benefit from using additional data sources such as LOD,[4] however not too much research was made in this area yet. One of the exception is work of Ostuni et al. [10] proposing graph based recommender system combining LOD knowledge and implicit feedback, or our own work on incorporating LOD directly as object attributes [13].

Interesting phenomenon is also development of user preferences over time. However no general conclusion was yet achieved: e.g. Xia et al.[14] uses time decay in item-to-item recommender systems, however e.g. Koren [7] argues that classical time-windows or instance-decay methods loses too much signal.

Among papers concerning recommending for e-commerce we would like to mention Linden et al. [9] describing Item-to-item collaborative filtering or from more recent work Belluf et al. [1] measuring business impact of recommender systems deployment in an on-line experiment.

3 Datasets

We have collected usage data from one of the major Czech travel agencies. Data were collected from December 2012 to April 2013. Travel agency is typical e-commerce enterprise, where customers buy products only once in a while (most typically once a year). The site does not force users to register and so we can track unique users only with cookies stored in the browser. User typically browses or searches through several

[3] `http://lastfm.com`, dataset available from `http://ir.ii.uam.es/hetrec2011/datasets.html`

[4] Linked Open Data, `http://linkeddata.org/`

categories, compares few objects (possibly on more websites) and eventually buys a single object. Buying more than one object at the time is very rare.

Our main objective while collecting dataset was to capture various possibly interesting data and thereafter decide about their usefulness in the experiments.

3.1 Implicit Feedback Data

In our previous work [11], only the user behavior on objects was monitored. Certain actions user committed was stored into database table to serve as implicit feedback. The table is in form of:

***ImpFeedback**(UID, OID, PageView, Mouse, Scroll, Time)*

UID and *OID* are unique user and object identifiers and Table 1 contains full description of implicit feedbacks. Note that *UID* is based on cookie stored by browser, so we cannot e.g. distinguish between two persons using the same computer. Table contains approx. 39 000 records with 0.09% density of *UID*× *OID* matrix.

Table 1. Description of the considered implicit feedbacks for user visiting an object

Factor	*Description*
PageView	Count(*OnLoad()* event on object page)
Mouse	Count(*OnMouseOver()* events on object page)
Scroll	Count(OnScroll() events on object page)
Time	Sum(time spent on object page)

3.2 Click Stream Data

However pages showing detail of an object represents less than 50% of visited pages. The rest consists mostly from various category pages accessed either via site menu or attributes search. The depth and broadness of category tree depends on the current user interface and nature of the objects of the domain. The travel agency website used in the experiments allows user to more or less freely combine values/intervals of several attributes (not all attributes of objects), where some predefined combinations are accessible via site menu and others can be derived through binary search. In order to determine importance of category pages for computing user preference, we have collected dataset containing user's click-stream throughout the website. The table is in form of:

***ClickStream**(UID, PageID, SessionNo, Timestamp)*

PageID serves as unique identifier of visited page. There is unique mapping from *OID* to *PageID*. ***ClickStream*** table contains approx. 121900 records and matrix *UID*×*PageID* has density of 0.17%.

3.3 Content Based Attributes

Finally each object and category page can be assigned with several content-based attributes. The information value of content-based attributes varies in different domains from very informative like e.g. laptops and computers to almost valueless like secondhand bookshops [13]. The travel agency dataset can be classified somewhere in between as there are some informative attributes, but a lot of important information is accessible only through textual description leaving some space for employing textual data mining techniques in the future. Table 2 contains list of available attributes for travel agency domain. In order to handle attributes properly in the experiments, they were transferred into the Boolean vector (Integer values e.g price was discretized equipotently into 10 intervals). The resulting ***Attributes*** matrix contains 2300 objects (and categories) with 925 features each.

Table 2. Description of content-based attributes and their cardinality per tour

Attribute	***Description***
TourType	Type of the tour (e.g. sightseeing)
Country	Destination country of the tour (e.g. Spain) [1..n]
Destination	More specific destination (e.g. Costa Brava) [0..n]
AccomodationType	Quality of the accommodation (e.g. 3*);
Accommodation	Specific accommodation for the tour [0..n]
Board	Type of board (e.g. breakfast, half-board)
Transport	Type of transport (coach, air…)
Price	Base price per person; integer
AdditionalInfo	IDs of information linked to the tour (e.g. about visited places, destination country etc.) [0..n]

4 Algorithms

Matrix factorization techniques are currently leading methods for learning user preferences, so we decided to adopt and slightly adjust them to suit our needs. We skip more elaborated introduction to the matrix factorization as it is quite well known technique and suggest Koren et al. [8] for more elaborated introduction.

Given the list of users $U = \{u_1,...,u_n\}$ and objects $O = \{o_1,...,o_m\}$, we can form the user-object rating matrix $\mathbf{R} = [r_{uo}]_{n \times m}$. With lack of explicit feedback, user-object rating r_{uo} in our case carries only Boolean information whether user *u* visited object *o*. For a given number of latent factors *f*, matrix factorization aims to decompose original **R** matrix into $\mathbf{UO}^T$, where **U** is $n \times f$ matrix of user latent factors (μ_i^T stands for latent factors vector for particular user u_i) and $\mathbf{O}^T$ is $f \times m$ matrix of object latent factors (σ_i is vector of latent factors for particular object o_i).

$$\mathbf{R} \approx \mathbf{UO}^T = \underbrace{\begin{bmatrix} \mu_1^T \\ \mu_2^T \\ \vdots \end{bmatrix}}_{n \times f} \times \underbrace{[\sigma_1 \quad \sigma_2 \quad \ldots]}_{f \times m} \tag{1}$$

Unknown rating for user i and object j is predicted as $\hat{r}_{ij} = \mu_i^T \sigma_j$. Our target is to learn matrixes **U** and **O** minimizing errors on known ratings. Regularization penalty is added to prevent overfitting. The optimization equation is defined as follows:

$$\min_{\mathbf{U},\mathbf{O}} \left\| \mathbf{R} - \mathbf{U}\mathbf{O}^T \right\|^2 + \lambda(\|\mathbf{U}\|^2 + \|\mathbf{O}\|^2) \tag{2}$$

This equation can be solved e.g. by Stochastic Gradient Descent (SGD) technique iterating for each object and user vectors:

$$\begin{aligned} \mu_i &= \mu_i + \eta \, (\sum_{j \in K_{ui}} (r_{ij} - \mu_i^T \sigma_j) \sigma_j - \lambda \mu_i) \\ \sigma_j &= \sigma_j + \eta \, (\sum_{i \in K_{oj}} (r_{ij} - \mu_i^T \sigma_j) \mu_i - \lambda \sigma_j) \end{aligned} \tag{3}$$

Where η is learning rate, K_{ui} set of all objects rated by user u_i and K_{oj} set of all users, who rates object o_j. The described method represents ***baseline*** algorithm. We now present three extensions to this method. Those extensions are independent of each other and can be combined freely.

4.1 Category Extension

Category extension expands list of objects to include also category pages covered in ***ClickStream*** dataset (see section 3.2). The rest of the algorithm remains the same. This allows us to better track movement of the user over the website and thus his/her preferences (note that both objects and categories may share some content attributes and thus we can infer their similarity).

Categories however cannot form full-bodied objects. One problem is that categories usually do not carry all features of the regular objects (e.g. there are some object attributes not available for binary search); however this is strongly domain dependent. The other problem is that categories cannot be effectively recommended, except maybe some very simple cases, as it would be impossible to derive any explanation for such recommendations (e.g. recommending category of Family Holidays in Spain in 2*hotels with half-board etc.). So the category objects may only aid us in inferring user preferences, not in fulfilling them.

4.2 Implicit Feedback Extension

Implicit feedback (***ImpF*** in experiments) extension involves deeper studying of user activity within the object and thus better estimating how much it is preferred. User activity is stored in ***ImpFeedback*** dataset (currently page views, time on page, mouse moves and scrolling events are monitored). Improved user-to-object rating r_{uo}^{+} replaces original Boolean r_{uo} where applicable.

The algorithm for computing r_{uo}^{+} was presented in our previous work [12]. It works in two steps, where various methods can be used in both steps. At first, it considers each feedback type (page views, time on page etc.) separately resulting into the vector of so called *local preferences* – real numbers from [0, 1] interval representing inferred user preference based on a single feedback type. In the second step, those values are aggregated together into a single value (r^{+}_{uo}).

In our previous work [12], we experimented with various procedures in both steps of the algorithm, but used rather simple recommending methods to test it. In our current work we focused more on usability of such information in up-to-date recommending methods and so used standard procedures here. We leave more detailed study of combination of both factors to our future work.

With the absence of any explicit feedback, we have adopted business-like approach and stated that buying an object represents full user preference. Preference based on other types of feedback is defined through its similarity with the purchasing behavior. Details of this approach and argumentation supporting it can be found in [12], in our current research was used weighted average to combine local preferences and discretized intervals of feedback type values in local preferences computation.

4.3 Attributes Extension

Attributes extension involves using content-based attributes of objects (and categories). We adopt approach of *Forbes and Zhu* [2] to deal with object attributes and implement their algorithm as PHP library. Their *content boosted matrix factorization method* is based on the assumption that each object's latent factors vector is a function of its attributes. Having $\mathsf{O}_{m\times f}$ matrix of object latent factors, $\mathbf{A}_{m\times a}$ matrix of object attributes and $\mathbf{B}_{a\times f}$ matrix of latent factors for each attribute, the constraint can be formulated as:

$$\mathbf{O} = \mathbf{AB} \tag{4}$$

Under the constraint (4), we can reformulate both matrix factorization problem (1), its optimization equation (2) and gradient descend equations (3):

$$\mathbf{R} \approx \mathbf{UO}^T = \mathbf{UB}^T\mathbf{A}^T = \underbrace{\begin{bmatrix} \mu_1^T \\ \mu_2^T \\ \vdots \end{bmatrix}}_{n\times f} \times \underbrace{\mathbf{B}^T}_{f\times a} \times \underbrace{\begin{bmatrix} a_1 & a_2 & \dots \end{bmatrix}}_{a\times m} \tag{1a}$$

$$\begin{aligned} \mu_i &= \mu_i + \eta\,(\sum_{j\in K_{ui}}(r_{ij} - \mu_i^T\mathbf{B}^T a_j)\mathbf{B}^T a_j - \lambda\mu_i) \\ \sigma_j &= \sigma_j + \eta\,(\sum_{(i,j)\in K}(r_{ij} - \mu_i^T\mathbf{B}^T a_j)a_j\mu_i^T - \lambda\mathbf{B}) \end{aligned} \tag{3a}$$

5 Experiments

In order to determine usefulness of additional data, we have examined each of their combination in an off-line experiment. We set the off-line evaluation as close to the real setting as possible, because we intent to deploy the recommender system into the full operation in the future.

5.1 Experimental Settings

We first needed to set experiment goals and success metrics. As the datasets contains only implicit feedback, we cannot rely on user rating and related error metrics e.g. RMSE or MAE (no need to mention, that those metrics do not reflect well real-world success metrics anyway). Precision / Recall methods are also problematical as the nature of observed implicit feedback is only positive[5] and absence of any feedback also cannot be automatically interpreted as negative feedback (user might not be aware of the object). As result, for arbitrary fixed user, we only have some evidence of positive preference for the minority of objects and know nothing about the rest.

Typical usage of recommender systems in e-commerce is to present a list of top-k objects to the user. We let recommending methods to rank objects and denote as success if the algorithm manages to rank well enough those objects, we have some evidence of their positive preference.

As we lack any explicit feedback, we need to infer positive preference from the implicit data. For the purpose of this rather early work we consider that every object the user has *visited* is positively preferred by him/her. It is possible to use more selective meanings of positive preference e.g. to consider only purchased objects as positively preferred. Such assumption will however lead to insufficient amount of data in the test set so we leave the problem of finer grained preference to the future work and bigger experimental datasets.

Recommending method evaluation was carried out as follows: For each user, his/her click stream was divided into two halves according to its timestamp – earlier data serving as train set and following as test set. Note that only users with at least two visited objects qualify for the experiment. There are other ways to divide train/test set e.g. to apply cross-validation, but we rather took advantage of possibility to use and compare stream or time-aware algorithms on the same dataset in future. The resulting train set contains 32480 records from 2956 users (13124 concerning objects and 19356 category pages). The test set contains 12749 records (objects only, as we intent to recommend only objects to the user).

All learning methods were initialized and trained with the same train set, 10 latent factors and maximal 50 iterations.

Then, for arbitrary fixed user, we let each method to rate all objects, sort them according to the rating and look up positions of objects from the test set. In production recommender system, we should take into account also other metrics like diversity, novelty or serendipity and probably want to pre-select the list of candidate objects, but for purpose of our experiment, we will focus on rating only.

We adopt normalized distributed cumulative gain (***nDCG***) as our main success metric. The premise of nDCG is that relevant documents appeared low in the

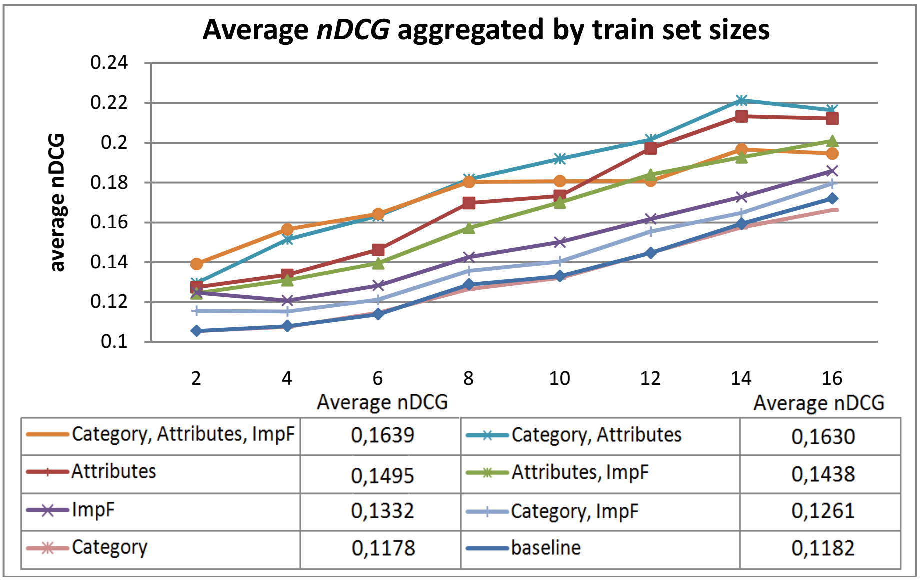
[5] Although some experiments and/or models of negative implicit feedback can be found in the literature, They are yet to be more extensively tested and eventually accepted.

recommended list should be penalized (logarithmical penalty applied) as they are less likely to attract user attention. This fits well for the recommending scenario, where lower-ranked objects are presented on less desirable positions. It is also possible to restrict DCG to sum only up to *top-k*th position as only *top-k* objects are shown to the user. However there is no justification to set any particular *top-k* and the list of eligible objects for recommendation could be pre-filtered (e.g. keep only objects from certain category if user is browsing the category). As result objects on lower ranks can keep some value too.

The results of ***Presence@top-k*** metric are also presented as it has more intuitive meaning. ***Presence@top-k*** for arbitrary fixed user is defined as quantity of preferred objects (objects from test set) within the *top-**k*** best objects list according to the prediction of current recommending method for current user. ***Presence@top-k*** is then summed over all users. ***Presence@top-k*** can quite well depict twists in methods performance over various top-k sizes.

5.2 Results

Figure 1 displays results of recommending methods in ***nDCG*** aggregated by the train set size and Figure 2 shows distribution of ***Presence@top-k*** up to top-100. Although smaller top-k would be used in the real deployment, the list of objects eligible for recommendation would be probably pre-filtered too, leaving some influence also to objects beyond typical top-k boundary.



	Average nDCG		Average nDCG
Category, Attributes, ImpF	0,1639	Category, Attributes	0,1630
Attributes	0,1495	Attributes, ImpF	0,1438
ImpF	0,1332	Category, ImpF	0,1261
Category	0,1178	baseline	0,1182

Fig. 1. Average nDCG aggregated by train set sizes per user. Legend shows average nDCG per all users and train set sizes

From the three aspects taken into account (data from categories, additional implicit feedback and objects attributes), using attributes of the objects have proven to be the most crucial. All methods with ***Attributes*** performed significantly better in terms of average nDCG than methods without ***Attributes*** (p-value $< 10^{-6}$, TukeyHSD test[6]). The price for this improvement is time complexity rising with the number of attributes.

Using additional data from category pages also improves recommendation, but only if accompanied with object attributes (p-value $< 10^{-6}$). This is quite easy to justify as only object attributes can sufficiently describe similarity between categories and corresponding objects.

The results are rather inconclusive about usage of other implicit feedback data. If used without ***Attributes***, they improved recommendations significantly (p-value $< 3.1*10^{-5}$), however using it together with ***Attributes*** did not improve results much further.

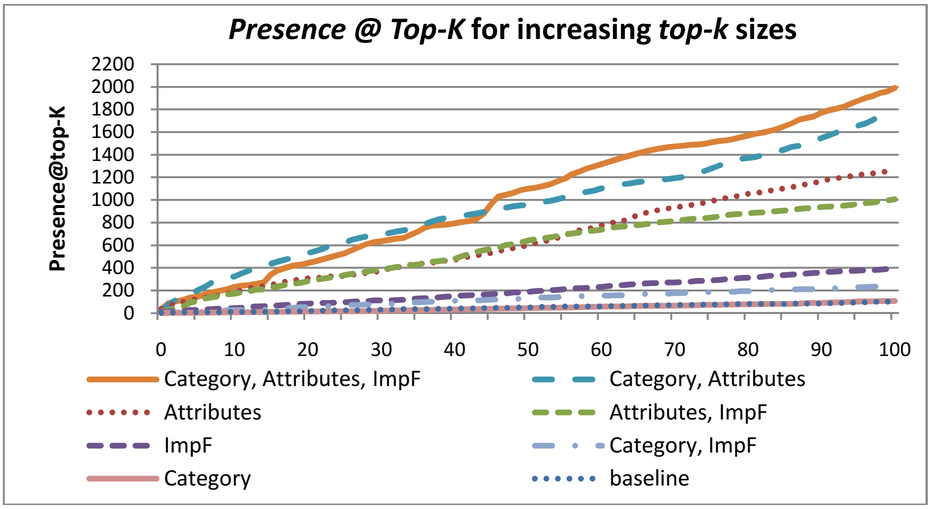


Fig. 2. Presence@top-k development for increasing top-k sizes

6 Conclusions and Future Work

In this paper, we were interested in the area of recommending for disloyal customers on e-commerce portals with low consumption rate. Our main task was to identify which data should be collected and how to use them to provide users with useful recommendations. We have identified three main extensions to common datasets: browsing history of categories, various implicit feedback types and content-based attributes of objects. We have adjusted contemporary matrix factorization techniques to handle such data and compare them in off-line experiment.

[6] Tukey Honest Significant Differences, `http://stat.ethz.ch/R-manual/R-patched/library/stats/html/TukeyHSD.html`

Both category browsing history and object attributes have proven to be worthy enhancement to the recommendation algorithms, however the results are not very conclusive about usage of more various implicit feedback features.

Future work involves e.g. experimenting with other recommendation methods or parameters of current ones. We would like to also consider other approaches to enhance matrix factorization with implicit feedback and object attributes and last but not least employing time-aware recommending algorithms. If experiments on other domains corroborate our ideas, our long-term goal is to deploy the system on real e-commerce portal and continue with on-line experimentation.

Acknowledgments. This work was supported by the grant SVV-2013-267312, P46 and GAUK-126313.

References

1. Belluf, T., Xavier, L., Giglio, R.: Case study on the business value impact of personalized recommendations on a large online retailer. In: RecSys 2012, pp. 277–280. ACM (2012)
2. Forbes, P., Zhu, M.: Content-boosted matrix factorization for recommender systems: experiments with recipe recommendation. In: RecSys 2011, pp. 261–264. ACM (2011)
3. Gauch, S., Speretta, M., Chandramouli, A., Micarelli, A.: User Profiles for Personalized Information Access. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web 2007. LNCS, vol. 4321, pp. 54–89. Springer, Heidelberg (2007)
4. Jawaheer, G., Szomszor, M., Kostkova, P.: Comparison of implicit and explicit feedback from an online music recommendation service. In: HetRec 2010, pp. 47–51. ACM (2010)
5. Kelly, D., Teevan, J.: Implicit feedback for inferring user preference: a bibliography. SIGIR Forum 37, 18–28 (2003)
6. Konstan, J., Riedl, J.: Recommender systems: from algorithms to user experience. UMUAI 22, 101–123 (2012)
7. Koren, Y.: Collaborative filtering with temporal dynamics. In: ACM SIGKDD 2009, pp. 447–456. ACM (2009)
8. Koren, Y., Bell, R., Volinsky, C.: Matrix Factorization Techniques for Recommender Systems. Computer 42, 30–37 (2009)
9. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing 7, 76–80 (2003)
10. Ostuni, V.C., Di Noia, T., Di Sciascio, E., Mirizzi, R.: Top-N recommendations from implicit feedback leveraging linked open data. In: RecSys 2013, pp. 85–92. ACM (2013)
11. Peska, L., Vojtas, P.: Evaluating Various Implicit Factors in E-commerce. In: RUE (RecSys) 2012. CEUR, vol. 910, pp. 51–55.
12. Peska, L., Vojtas, P.: Negative Implicit feedback in E-commerce Recommender Systems. In: Proc. of WIMS 2013, pp. 45:1-45:4. ACM (2013)
13. Peska, L., Vojtas, P.: Enhancing Recommender System with Linked Open Data. In: Larsen, H.L., Martin-Bautista, M.J., Vila, M.A., Andreasen, T., Christiansen, H. (eds.) FQAS 2013. LNCS (LNAI), vol. 8132, pp. 483–494. Springer, Heidelberg (2013)
14. Xia, C., Jiang, X., Liu, S., Luo, Z., Yu, Z.: Dynamic item-based recommendation algorithm with time decay. In: ICNC 2010, pp. 242–247. IEEE (2010)