Nameling Discovery Challenge - Collaborative Neighborhoods

Dirk Schäfer and Robin Senge

Mathematics and Computer Science Department
Philipps-Universität Marburg, Germany
dirkschaefer@jivas.de, senge@informatik.uni-marburg.de

Abstract. This paper describes a series of experiments designed to solve the "Nameling" challenge. In this task, a recommender should provide suggestions for interesting first names, based on a set of names in which a user has shown interest. An approach based on dyadic factors is proposed where side-information about names and users were incorporated. Furthermore, factors based on User-based Collaborative Filtering play a central role. The performance considering the neighborhood and binary similarity measures was assessed.

Keywords: collaborative filtering, implicit feedback, dyad, competition

1 Introduction

Implicit feedback data can be collected whenever users are interacting with information systems. In connection with recommender systems this data is appealing, because it is obtainable in large quantities, e.g. from log files, and can be used as a complementary information source to rating data. On the one hand, the popularity of this kind of data is reflected by the numerous synonyms[1] in literature [6, 5, 12, 11]. One the other hand, there are various applications ranging from basket case analysis [7] to large scale news recommendation [2] and applied machine learning fields, e.g. Information Retrieval and Recommender Systems research to name a few.

2 The Challenge

The Nameling discovery challenge is part of a workshop held at the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases[2] 2013.

[1] 0-1 data, one-class data, positive-only feedback, click-stream data, market basket data, click-through data
[2] ECML PKDD

2.1 Task Description

Given is a set of names that has been recorded as input by users of the Nameling web platform [8]. The task consists in recommending further names for a subset of selected users. The recommender should build an ordered list of one thousand names for each test user, where the most relevant names are placed at high rank positions. As performance metric the Mean Average Precision (MAP@1000) had to be used.

Table 1. Notations used in the paper

Symbols	Definition
$\mathcal{U}$	Set of all users
$\mathcal{I}$	Set of all items
u,v	Indices for users
i,j	Indices for items (=names)
$\mathcal{I}$(u)	Item set of user u
$\mathcal{U}$(i)	Set of users that have an affiliation with item i
w_{uv}	Similarity between two item sets
s_{uj}	Propensity for user u to select item j
$\mathcal{N}$	Items that are listed in the file "Namelist.txt"

2.2 Dataset

The dataset contains a training set which is a sparse matrix consisting of 59764 users (rows) and 17479 names (columns) and a test set with 4140 user IDs. Furthermore, a namelist file[3] is provided consisting of 44k names. For each user of the test set, two names from the system activity "ENTER_SEARCH", that are also contained in the namelist, have been extracted and are held out by the challenge organizers. Two Perl scripts were provided, the first one is able to build an own training and test set with names from the challenge training set exactly the same way as the challenge training and test set were built. The second script can be used for evaluation. In Figure 1 the sparsity of the training set is reflected by the frequencies of the most often chosen names. It shows a long-tail distribution, i.e. a small amount of names is very popular and at position 2000 names occur, which were only chosen by few users.

3 Related Work

In recommender systems literature, algorithms are classified as either being neighborhood-based (aka memory-based) or model-based. In the first group, there are user-based and item-based collaborative filtering methods. For user-based k-nearest neighbor collaborative filtering (UCF) the idea is to identify a

[3] Namelist.txt, see abbreviation $\mathcal{N}$ in Table 1

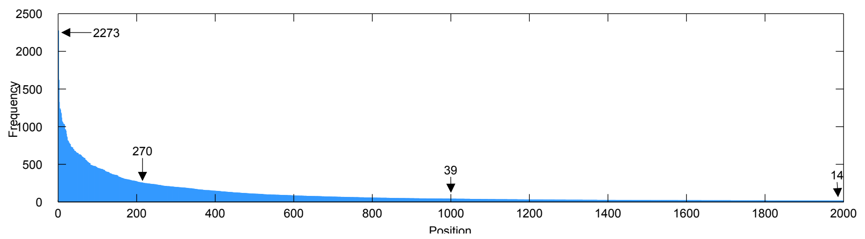

Fig. 1. Frequencies of top 2000 names.

group of k most similar users for each user to infer a ranking of items that are new for the user and also most interesting. Item-based nearest neighbor CF in contrast identify new items that have a relation to the existing item set of a user. The item-based CF approach has the advantage that a recommendation is easily explainable to the user and much more efficient to generate compared to UCF. On the other hand it lacks in accuracy.

For the model-based approaches and especially for this setting of implicit feedback, various methods based on Matrix Factorization (MF) have been created. In the One-Class Collaborative Filtering framework from Pan et al. two extreme assumptions about the data are being made [11], these are, that all missing values are all negative (AMAN) and all missing values are unknown (AMAU). They use weighted matrix factorization and sampling strategies to cope with the uncertainty about the unobserved data.

Another MF method that deals with side-information has been proposed in [4], where an embedding of auxiliary information into a weighted MF has been proposed. In [12] the Bayesian Personalized Ranking framework for implicit feedback has been described where also MF in combination with a bootstrap sampling approach is used to learn from pairwise comparisons.

NameRank is an item-based recommendation approach that has been proposed recently in combination with the Nameling data set and showed very good performance compared to the above mentioned approaches [10].

4 Recommendations Using Various Dyadic Factors

To begin with, we describe how dyadic factors can be used to induce rankings on names. The rest of the section deals with the engineering of these factors with two different approaches. The first one aims at improving the most popular items approach by using side-information for users and names, whereas the second approach is based on Collaborative Filtering.

4.1 Basic Scoring Scheme

In the following, each user-item pair (a dyad) receives a score based on the following equation:

$$s_{ui} = d_{ui} \cdot f_{u,i}^{[1]} \cdot f_i^{[2]} \tag{1}$$

Sorting s_{ui} scores in descending order for user u provides a ranking of items that can be recommended. The formula consists of a dyadic factor d_{ui} and two filter factors. The filters are indicator functions that adress the requirements of the prediction task. $f^{[1]}$ is 0 for names that are members of the user item-set $I(u)$. And the purpose of the other filter $f^{[2]}$ is to comply with the demand to accept only names that are part of the namelist $\mathcal{N}$. The various possibilites to engineer the dyadic factor are described below.

4.2 Discovering the "Most Popular" Baseline

In first experiments we found out that by simply considering a constant top 1000 majority vote for all names[4], we were already able to outperform some of the results others contributed to the leaderboard at an early stage. We could even improve this result by considering the 2000 top items and filtering out the training names of the individual users, which lead to the definition of the filter factor $f^{[1]}$. It turned out, that recommending items that way is known in literature as the Most Popular (MP) approach [10]. In the course of the challenge, we could identify other teams that scored equally, and we think they recommended names using the same approach. Because of this and its simplicity we say we discovered the "baseline method" within the leaderboards.

4.3 Side-Information on Names

In the given data set, there were no additional attributes on the item objects provided. Therefore we constructed two features as side-information for names: length of name and gender. Both features are characterized by having a finite set of discrete values as co-domain. With that, the now explained general procedure is applied, where the Most Popular Ranking is used as input. The idea is to split the MP Ranking into several queues corresponding to the available discrete values of a feature and later to adjust the MP recommendations for some of the users on basis of their item sets. From the queues, items are sampled according to proportions found in the item sets of the user. Since the item sets are typically small (see Figure 2), some significance criteria have to be met, e.g. a minumum size of the item-sets. Having found a ranking of at least 1000 names that way, we turned the ordered list of names in to numerical factors by assigning score values uniformly according to the rank position in an interval [max, min], e.g. the top ranked names recieved scores of $1, 0.9, 0.8, \ldots$[5].

Name Length Factor The general strategy of sampling from queues described above had to be slightly adapted due to the fact that there are 13 discrete values for name lengths, but item sets of users are too small to capture the proportions sufficiently (see Figure 2). To be able to sample from all those queues, the user

[4] that were used by the approx. 60k training users
[5] we actually used the interval [1,0.1]

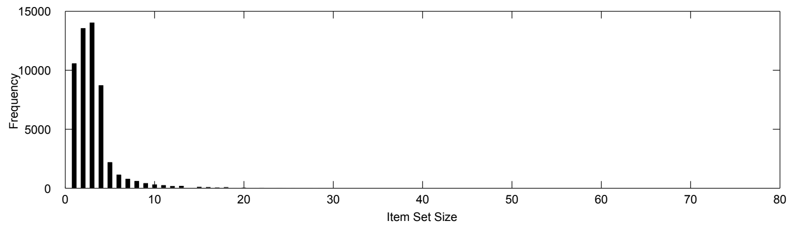

Fig. 2. Item set sizes across the training data

preferences for items, in this case for short or long names, must be significantly explainable. For this reason we decided to join the discrete values in two almost equally balanced sets: the set of shorter names (length 1 to 5) and longer names (length 6 to 13), see Table 2.

Table 2. Frequencies according to different lengths of names

Length of Name	1	2	3	4	5	6	7	8	9	10	11	12	13
Frequency	7	6	98	354	522	455	277	158	85	25	8	3	1

Gender Factor Since it was officially allowed and explicitly encouraged to use additional data sources for the offline challenge, we decided to include gender information for the names. By doing this, the Most Popular performance could be considerably improved (see Table 5). We extracted all names from the training set which were associated with the activity "ENTER_SEARCH" and used an external program[6] to classify names into the following three classes: 0 - unisex name or not identifiable, 1 - male, 2 - female. We used the idea described above and sampled from three queues according to the item sets' gender distributions until the new recommendation set reached a size of 1000 names.

4.4 Side-Information on Users

In previous uses of side-information, **one** global Most Popular recommendation was taken as basis and according to item set statistics the recommendation was modified by reordering or filtering. In further experiments, we followed a different approach: for groups of users, specified by their attributes, **many** Most Popular recommendation lists are created. Approximate geo locations of users were provided that were derived from IP addresses. We used in particular the country information for defining a new factor as described next.

[6] gender.c by Jörg Michael, which has been cited in the German computer magazine c't 2007: Anredebestimmung anhand des Vornamens.

Demographic Factor Additional geographical information was available for a subset of users. For every country we created a separate Most Popular recommendation list (see Table 3 for an excerpt). Unfortunately, the performance could only be increased marginally by this effort (see Table 5).

Table 3. The most popular names for seven countries are listed. The ? symbol represents the group of users that could not be geo-located.

Country	Frequency	Names						
?	36639	emma	anna	julia	michael	paul	thomas	christian
DE	19411	julia	emma	anna	michael	greta	emil	alexander
AT	1714	emma	paul	andreas	michael	anna	alexander	katharina
CH	661	max	sandra	christian	anna	daniel	katharina	jan
US	350	laura	sophie	emma	august	sarah	johann	leo
GB	134	matilda	emma	pia	martin	caroline	anja	robert
FR	99	sebastian	alma	simon	solveig	emma	lisa	emil
IT	79	emma	astrid	katharina	sophie	johanna	ida	verena

4.5 The User-based Collaborative Filtering Factor

User-based collaborative filtering is known as a successful technique to recommend items based on the **ratings** of items by different users. For this technique, the choice of a similarity measure that captures how similar users are, and the choice of neighborhood are the most important factors. In order to cope with this kind of binary data, the similarity has to be chosen suitably for implicit feedback data. And, as will be shown later, also the neighborhood size is even more crucial for the performance. The general user-based CF formula provides a score for each user-item pair as follows:

$$p_{uj} = \kappa \sum_{v=1}^{|\mathcal{U}|} w_{uv} c_{vj} \tag{2}$$

with indicator function

$$c_{ui} = \begin{cases} 1, & \text{if } i \in \mathcal{I}(u) \\ 0, & \text{else} \end{cases}$$

and a normalization term

$$\kappa = \frac{1}{\sum_{v=1}^{|\mathcal{U}|} w_{uv}}$$

to ensure that $p_{uj} \in [0, 1]$. Note that Equation (2) shows a special case, where the similarities of user u to all other users v are considered. In literature often only a neighborhood around u of the top N similar users are taken into account.

Similarity Measures A classical similarity measure between two vectors consisting of binary numbers is the Jaccard Index, which is the proportion between the sizes of the intersection and the union of two sets.

$$w_{uv} = \frac{|\mathcal{I}(u) \cap \mathcal{I}(v)|}{|\mathcal{I}(u) \cup \mathcal{I}(v)|} \tag{3}$$

In this context, it means two item-sets are more similar the more common items they share.

Good and often superior results were reported regarding the "log-likelihood similarity" which is implemented in the Mahout software package[7] for item- and user-based collaborative filtering [10, 3]. In computational linguistics Dunning proposed the use of the likelihood ratio test to find rare events as alternative to traditional contingency table methods [3]. In a bigram study he aimed at finding pairs of words that occurred next to each other significantly more often than expected from pure word frequencies. As basis for the log-likelihood statistics he used the following contingency table (see Table 4).

Table 4. Contingency table as used for the Dunning similarity. The first table shows the generic structure and below is the same table with the actual values in the UCF context.

	Event A	Everything but A
Event B	A and B together(k_{11})	B, but not A(k_{12})
Everything but B	A without B(k_{21})	Neither A nor B(k_{22})

	Event A	Everything but A
Event B	$\lvert\mathcal{I}(u) \cap \mathcal{I}(v)\rvert$	$\lvert\mathcal{I}(v)\rvert - \lvert\mathcal{I}(u) \cap \mathcal{I}(v)\rvert$
Everything but B	$\lvert\mathcal{I}(u)\rvert - \lvert\mathcal{I}(u) \cap \mathcal{I}(v)\rvert$	$\lvert\mathcal{I}\rvert - \lvert\mathcal{I}(u)\rvert - \lvert\mathcal{I}(v)\rvert + \lvert\mathcal{I}(u) \cap \mathcal{I}(v)\rvert$

The Dunning similarity, as we call it, is the log-likelihood ratio score defined[8] as follows:

$$w_{uv} = 2[H(k_{11} + k_{12}, k_{21} + k_{22}) + H(k_{11} + k_{21}, k_{12} + k_{22}) - H(k_{11}, k_{12}, k_{21}, k_{22})]$$

where $H(X)$ is the entropy defined as

$$H(X) = -\sum p(x) \log p(x).$$

Both similarity measures can be characterized regarding their use of information from two binary vectors under consideration. In [1] a survey of 76 binary similarity and distance measures had been carried out. All measures were described by a table called Operational Taxonomic Units that is identical to the contingency table shown above in Table 4. The measures fall in two groups regarding their use of negative matches that corresponds to the cell value k_{22}. The Dunning

[7] http://mahout.apache.org/ - scalable machine learning libraries
[8] for implementation details refer to the Appendix

similarity is not covered in that survey but falls into the category of similarity measures that make use of that kind of information. In contrast, the Jaccard similarity belongs to the negative match exclusive measures. The general inclusion or exclusion of negative matches has been debated over years and the particular choice of a measure is surely domain dependent.

4.6 Results

The MAP performances for the dyadic factor approaches based on MP are given in Table 5. Among those, the dyadic factor using user geo-locations performs best. Otherwise, the dyadic factor approaches based on UCF perform clearly better, if neighborhood sizes of UCF factor p_{ui} are larger than 500 (see Figure 3). The best performance values can be achieved when choosing the neighborhoods as large as possible. This means to consider all users according to Equation (2). To conclude, the choice between Dunning and Jaccard similarity for the UCF factor is not that relevant. In contrast, the neighborhood size is much more important. However, if not many users are available in the system and furthermore not much is known about users and items, the MP approach then provides a solid basis.

Table 5. MAP performance values for Most Popular experiments.

Method	Most Popular	Length of Name	Gender	Country
MAP	0.0247	0.0248	0.0252	0.0256

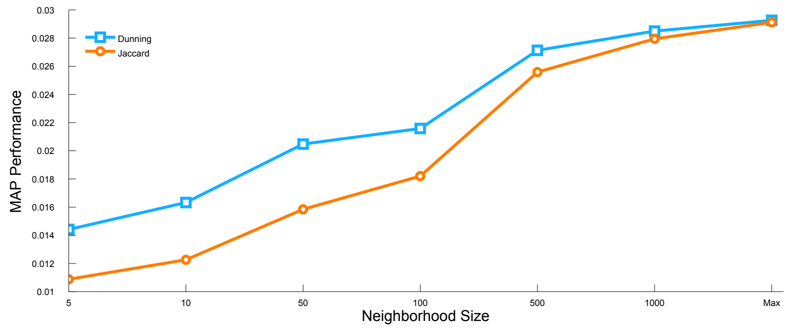

Fig. 3. Dyadic factor approach with different neighborhood sizes and two different similarity measures for the UCF factor. Mean MAP values measured on 50 test set splits of size 1000.

5 Hybridization by Combining Multiple Dyadic Factors

We propose to combine the different dyadic factors presented in the last section by extending the basic scheme. Furthermore, experimental results are shown for different factor combinations.

5.1 Extended Scoring Scheme

In this section, we show how different dyadic factors can be combined into a unified scoring scheme, which is an extension to Equation (1).

$$s_{ui} = D_{ui}^{\Theta} \cdot f_{u,i}^{[1]} f_i^{[2]} \tag{4}$$

$$D_{ui}^{\Theta} = (d_{ui}^{[1]})^{\gamma_1} \cdot (d_{ui}^{[2]})^{\gamma_2} \cdot \ldots \cdot (d_{ui}^{[m]})^{\gamma_m} \tag{5}$$

The choice of a dyadic factor combination is governed by the hyperparameter set $\Theta = \{\gamma_1, \gamma_2, \ldots\}$, where each parameter has the purpose of weighting the contribution of a particlar dyadic factor to the score s_{ui}.

5.2 Optimization

The optimization of the hyperparameter set Θ for formula (4) for $\forall u \in \mathcal{U}$ and $\forall i \in \mathcal{I}$ to maximize the overall MAP value is not trivial, because gradient based methods are not applicable here. For this reason we used grid search and an evolutionary algorithm[9] to find a well weighted combination of dyadic factors.

5.3 Experiments and Results

For a random selection of 1000 test users the following dyadic factors described in the last section were considered:

- The demographic factor c_{ui}, where the most popular items are recommended according to the country of a user.
- The length of a name factor n_{ui}.
- The gender of a name g_{ui}.
- User based Collaborative Filtering factor using Jaccard similarity p_{ui}.

According to Table 6 the combination of multiple dyadic factors can be beneficial. However, regarding the MAP performance only minimal improvements can be observed.

[9] CMA-ES from Apache Commons (http://commons.apache.org).

Table 6. MAP performance values for different combinations of factors

D_{ui}^{Θ}	Hyperparameters	MAP	Best Single Factor
$c^{\gamma 1} n^{\gamma 2}$	1.751, 0.066	0.0261	c_{ui} (0.0257)
$c^{\gamma 1} g^{\gamma 2}$	1.082, 0.941	0.0255	c_{ui} (0.0257)
$p^{\gamma 1} c^{\gamma 2}$	1.137, 0.516	0.0350	p_{ui} (0.0350)
$p^{\gamma 1} c^{\gamma 2} n^{\gamma 3} g^{\gamma 4}$	2.449, 1.365, 0.726, 0.877	0.0356	p_{ui} (0.0350)

6 Discussion

During the challenge phase we could confirm findings reported in literature: The baseline method described in section 4.2 performs well compared to statical methods using relational data from co-occurrence networks [9]. We found out that the choice of input for the recommendation has a large impact on the performance, e.g. restricting the item sets for the UCF approach a-priori to names contained in $\mathcal{N}$ has a negative effect. Furthermore, we introduced an approach based on weighted dyadic factors, which enabled us to combine different information sources and assumptions in a formal way. That allowed us to express the preferences of users by different factors and provides further possibilities, that are out of the scope of this paper, e.g. to combine User-based with Item-based Collaborative Filtering. Even though this approach seems to be appealing at first sight, the MAP performance improvements are only minimal compared to single dyadic factors. Introducing the various dyadic factors using side-information, they performed not as well as factors based on Collaborative Filtering, which shows the effectiveness of UCF despite its simplicity.

References

1. Seung-Seok Choi, Sung-Hyuk Cha, and C Tappert. A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, 8(1):43–48, 2010.
2. Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, pages 271–280. ACM, 2007.
3. Ted Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational linguistics*, 19(1):61–74, 1993.
4. Yi Fang and Luo Si. Matrix co-factorization for recommendation with rich side information and implicit feedback. In *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, pages 65–69. ACM, 2011.
5. Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 176–185. IEEE, 2010.

6. Michael Hahsler. Developing and testing top-n recommendation algorithms for 0-1 data using recommenderlab, 2010.
7. Andreas Mild and Thomas Reutterer. An improved collaborative filtering approach for predicting cross-category purchases based on binary market basket data. *Journal of Retailing and Consumer Services*, 10(3):123–133, 2003.
8. Folke Mitzlaff and Gerd Stumme. Namelings - discover given name relatedness based on data from the social web. In Karl Aberer, Andreas Flache, Wander Jager, Ling Liu, Jie Tang, and Christophe Guret, editors, *SocInfo*, volume 7710 of *Lecture Notes in Computer Science*, pages 531–534. Springer, 2012.
9. Folke Mitzlaff and Gerd Stumme. Relatedness of given names. *Human Journal*, 1(4):205–217, 2012.
10. Folke Mitzlaff and Gerd Stumme. Recommending given names. *arXiv preprint arXiv:1302.4412*, 2013.
11. Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. One-class collaborative filtering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 502–511. IEEE, 2008.
12. Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461. AUAI Press, 2009.

Appendix

Calculation of the Entropies for the Dunning Similarity

The entropies for the Dunning similarities using the notation of Table 4 can be calculated as follows :

$$LLR = 2[H(row) + H(col) - H(row, col)]$$

$$H(row) = -((k_{11} + k_{12}) \log(k_{11} + k_{12}) + (k_{21} + k_{22}) \log(k_{21} + k_{22}))$$

$$H(col) = -((k_{11} + k_{21}) \log(k_{11} + k_{21}) + (k_{12} + k_{22}) \log(k_{12} + k_{22}))$$

$$H(row, col) = -((\sum_{ij} k_{ij} \log(\sum_{ij} k_{ij}) - (\sum_{ij} k_{ij} \log k_{ij})).$$