# Indigo Books and Friends Recommendation

Arnold Binas, Laurent Charlin, Alex Levinshtein, and Maksims Volkovs

Department of Computer Science
University of Toronto

April 16, 2009

**Abstract**

In this report we take first and significant steps towards augmenting Indigo's online presence by automatically personalized features that can be expected to help online sales: book as well as online community friends recommendations. We use and modify well-known collaborative filtering techniques to utilize a single machine learning method for both tasks. Our techniques are shown to be effective both quantitatively and by data visualization, and additionally yield valuable insights into how a refined data acquisition strategy may yield even more valuable results for Indigo.

# Contents

# 1 Introduction

**Abstract**

The motivation of the project is to increase online sales by providing personalized recommendations of both books and potential book interest-based friends to users. We take a collaborative filtering approach that leverages the wisdom of the crowd in their collective book ratings data to achieve this goal.

In this report we summarize our work on the Indigo book and friend recommendation project, undertaken as part of the CSC2125 consulting course at the University of Toronto. We outline high-level and motivational issues, but also go into some technical detail about our methods and results to provide the client with a sufficient basis for continuing and implementing this project as part of their online operations. To aid selective reading of parts of this report, every section is preceded by a concise summary.

The report is organized as follows. After giving a brief introduction about the project, we introduce the dataset in Section 2. Section 3 introduces the collaborative filtering rating prediction method that underlies both our book and friends recommendation techniques. The techniques and results for book recommendation are treated in Section 4 and for friends recommendation in Section 5. We summarize our proposed solution and offer a view on challenges and suggested future work in Section 6.

## 1.1 Motivation

The long-term motivation of this project is to increase Indigo's online product sales. This can be done either directly or encouraged indirectly.

A direct approach to driving online sales is to provide live personalized book recommendations to chapters.indigo.ca account holders. In fact, a study led by ChoiceStream shows a strong monetary incentive to provide recommendation to users [1]. They show that 78% of users are interested in receiving personalized content and that almost half of users "are more likely to shop at sites that provide personalized recommendations".

Indirect approaches involve making the members' online experience more engaging, encouraging members to spend more time on the site, and thus increasing the probability of sales. In the case of Indigo's online community members, one powerful way of increasing member engagement is to encourage community friendships based on common book interests. In particular, by suggesting to each member a set of possible friends, the member is encouraged to interact with other like-minded members. In the medium- to long-term, such interactions have the potential to augment the member's interests in new books and hence yield additional sales. In addition, a lively online community could lead to increased visibility of Indigo's online presence to search engines, yielding interest-driven traffic to the website.

## 1.2 Goals and methods overview

The goal of this project is to provide the technical foundations for realizing one method each for directly and indirectly encouraging online sales. In particular, we are concerned with improving personalized book recommendations to spawn members' interest as well as creating friends recommendations to encourage a lively and interacting online community.

### 1.2.1 Book recommendation

Providing good recommendations starts with accurately modeling a customer's preferences toward products. Ratings, compared to purchase history or virtual product shelves, are one of the most expressive means by which users are allowed to express their product preferences. Different from the purchase history, customers can rate books that they have not bought through Indigo's online store.

Accordingly, in this project we concentrate on generating personalized book recommendations from the ratings data with a brief look at how some of the other data sources might be used to supplement the ratings. We will see that in order to recommend books to users, a first step is to learn to predict their book preferences in terms of ratings.

### 1.2.2 Friends recommendation

The goal of this part of the project is to recommend Indigo community members other members for potential friendship on the basis of common book interests. Here we follow a related approach to the one taken for book recommendations and focus our efforts on the book ratings data. As we will see in later sections, the ratings and other data allows us to extract a similarity measure between users that is based on their respective book ratings, i.e. reflects the users' book interests. The extraction of informative user and item (book) descriptors through rating prediction from large ratings datasets is commonly done by *collaborative filtering*, introduced on a high level below.

## 1.3 Collaborative filtering approach

The term collaborative filtering refers to the extraction of information about individuals from large datasets concerning a large number of individuals. In the case of this project, our primary dataset is the set of ratings for all Indigo community members and we are interested in learning about individual users' book preferences (both for book and friends recommendation). Compared to the huge amount of books available at Indigo, each user will only rate a tiny number of books. But there is also a very large number of users, many with different book interests, so between all users, most books will have been rated by several users. For many users, some of the books they rate will overlap with other users' collection of rated books. The idea of collaborative filtering is then to leverage the "wisdom of the crowd", and through the large number

of user-book rating data, induce a book-preference function for each user-book pair. Several methods exist for doing collaborative filtering; we introduce the one that we use in this report in Section 3.

# 2 The Data

**Abstract**

A diverse set of potential data sources is available, but their diversity and sparsity presents significant challenges. The most expressive type of data are the book ratings by users, and we will concentrate on this data source here.

## 2.1 Description

We were provided with several portions of Indigo's community member data, which we describe in some detail in this section. Some types of data are available for all Indigo online customers, while other types of data exist only for community members. In this project, we focus on recommending books and friends to community members, so all types of data can in principle be used. Below we first summarize the available data before discussing some of the challenges that it presents.

### 2.1.1 Book ratings

The book ratings data simply consists of triples $(i, j, R_{ij})$ for user $i$ rating book $j$ with rating $R_{ij}$. Book ratings data is available for every community member. It can be viewed as a matrix $R$ of $N$ rows and $M$ columns, where $N$ is the number of users and $M$ the number of books, which only contains entries for the $(i, j)$ for which rating data is available. Some statistics about this ratings matrix are as follows.

| | |
|---|---|
| Total number of ratings: | 296,031 |
| Number of users who rated at least one product: | 25,563 |
| Number of products rated by at least one user: | 86,598 |
| Matrix density: | 0.000134 |
| Mean number of ratings per user: | 11.58 (std: 32.96) |
| Mean number of ratings per product: | 3.42 (std: 14.62) |
| Mean rating: | 4.16 (std: 0.99) |

Note that the matrix is extremely sparse (very low density). In fact, it is two orders of magnitude sparser than the well-known Netflix challenge dataset. Figure 1 additionally shows histograms characterizing the ratings data. One important thing to note here is the skewedness of the ratings distribution: People tend to rate books they like much more frequently than those that they do not like. This is in stark contrast to the "true" ratings distribution: If any given user were asked to rate all books in existence, it is reasonable to assume that only a small fraction would receive high ratings. The distributions of the number of ratings per user and product are also skewed. Many of the total number of ratings are concentrated on a few well-known books (this is a good example of

a long-tail distribution). Likewise, there are a few users who rated many books, but most users did not rate many at all.

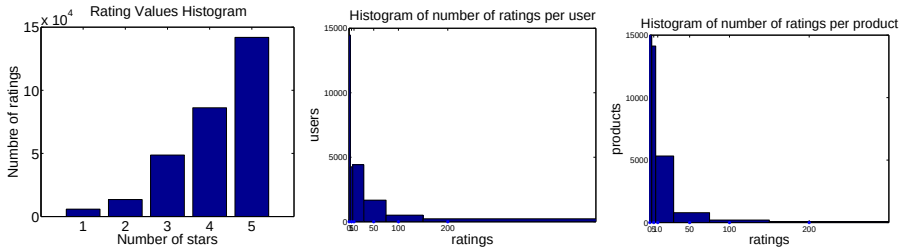


Figure 1: Left: distribution of ratings given; center: histogram for the number of ratings per user; right: histogram for the number of ratings per item

### 2.1.2 Purchase histories and virtual book shelves

Purchase history and virtual bookshelf items are two additional potentially rich sources of information about users' book preferences. The purchase history is simply a list of books recently purchased online for each user. Bookshelf items are books that community members have placed on their virtual bookshelves, indicating a certain preference for these books. It is important to note that this relation is one-sided (i.e., nothing can be inferred about a book that is not on one of the user's shelves). The purchase history is in principle available for every Indigo online customer, while virtual bookshelves are a feature of the online community and are thus only available for its members.

Figure 2 shows the distribution of the number of shelf and puchased books per user, respectively. Again a similar pattern as for the rating data can be observed. A few members have many books on their shelves, or have bought many books in the recent past, while for most members only very few data points are available.

### 2.1.3 Friends

The online community social networking feature allows users to add other members as friends. This friends data was also available for this project and can be thought of as a binary square matrix containing 1's for every pair of members who are friends and 0's for all other pairs. The intuition of the client was that the current pattern of friendships in Indigo's online community is based on real-world friends as opposed to being based on common book interests. It is this latter kind of friendship that we would like to induce by the recommendations in this project. Hence we do not use the friendship data for generating recommendations here, but we do use it to test the hypothesis that it does not reflect common book interests.

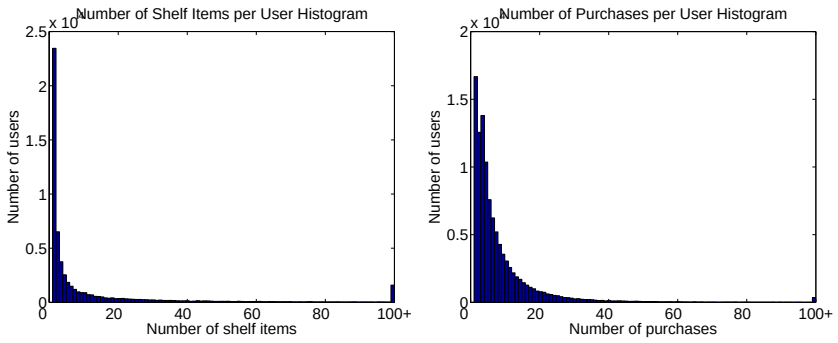


Figure 2: Left: histogram for the number of shelf items per user; right: histogram for the number of recent purchases per user

### 2.1.4 Additional information

Several other sources of information are available that may or may not be helpful in recommending books or friends. We list here types of data that we did not exploit for either task as that would have been outside the scope of this project, but some of the following may add value if appropriate methods can be developed for their inclusion.

Side information is available for both books and members. Books have titles, authors, and sometimes categories. Much of the collaborative filtering community believes that given a dense enough ratings dataset, such additional side information cannot contribute to better rating prediction performance. We have collected the fine-grained book categories into super categories for visualization. Users have names, are marked as being an Indigo employee or iRewards member, and their rough geographical location is available in the form of their postal code. The purchase history also has side information beyond the mere book identifiers that we use in this project, namely timestamps that may contain additional useful information.

Yet more information is available for those members that are members in Indigo's online community, i.e. the type of members that we focused on in this project. Community members can write book reviews, make recommendations to other members, devise top 10 lists of favorite books, and join groups within the social network. The latter may be valuable for the friends recommendation task.

## 2.2 Challenges

The data described in this section presents a number of challenges. We have already mentioned that the ratings, our primary dataset, are heavily skewed towards high values. This makes it harder for a machine learning algorithm to learn the true distribution of ratings, and the true ordering of books by

members' interests, because the algorithm sees far more positive than negative ratings.

Another challenge of the data is that there is simply not a lot of it compared to the number of members and books in the database. The statistics in Section 2.1.1 illustrate this fact for the ratings data, but the shelves and purchase history data are similarly sparse. The problem here is that while there are a few members with many ratings each, the vast majority of members has rated only very few if any books, making it difficult to learn their book preferences.

Another challenge, and also opportunity, is the fact that the total amount of data comes from several difference sources (ratings, shelves, purchases, ...). This means that more than one source of information is available for learning members' book interests (the opportunity), but also makes simple plug-and-play algorithms inapplicable (the challenge).

# 3 Rating Prediction by Collaborative Filtering

**Abstract**

The underlying machine learning problem we solve in order to generate both book and friends recommendations is to predict ratings for books that a given user has not yet rated. The algorithm we choose to address this problem produces book and user descriptors that can be used to characterize and compare both types of objects.

We introduced the underlying idea of collaborative filtering in Section 1.3. In this section, we will focus on the specific collaborative filtering method used for rating prediction in this project. We first motivate the use of a rating prediction method for our goals of book and friends recommendation. We then give a naive baseline method compared to which we can measure the merit of the more sophisticated probabilistic matrix factorization (PMF). Last we introduce the PMF method in some detail.

## 3.1 Rating prediction

Rating prediction is the task of using existing book ratings to predict users' ratings on all other (not-yet-rated) books. To measure success in this task, a set of known ratings can be discarded from the training set (on which the method bases its predictions) and used to check predicted ratings against.

Given a rating prediction method, it can be used to make book recommendation by simply ordering all books for a given user by their predicted rating by that user and recommending the top-rated books. However, any rating predictor alone is not enough to yield friend recommendations. In Section 3.3, we will see that our chosen rating prediction method learns features useful for friend recommendation simultaneously, making it especially appealing for this project.

## 3.2 Naive baseline method

The most naive method for rating prediction that is better than random guessing assigns ratings to books independent of the user. I.e., for every user, the predicted rating for a given book is the same. The baseline method works as follows. Let $I_{ij}$ be 1 if user $i$ rated book $j$ (and 0 otherwise) and let $R_{ij}$ be the rating the user assigned to the book if it was rated. We can then average all ratings for book $j$ over all users $i$ which have rated the book:

$$R_j^* = \frac{1}{N_j} \sum_i I_{ij} R_{ij} \tag{1}$$

where $N_j = \sum_i I_{ij}$ is the number of users who have rated book $j$. $R_j^*$ is the predicted rating for book $j$, same for each user. Rating predictions using this averaging method thus correspond to the overall popularity of books. Sorting all books by their predicted global ratings then yields an order in which books can be recommended by popularity, but this recommendation order is the same

for every user. In addition to this shortcoming, this simple baseline method cannot be used to recommend users to each other as friends based on their book interests.

## 3.3 Probabilistic matrix factorization

### 3.3.1 The method

Probabilistic matrix factorization (PMF) was introduced by researchers at the University of Toronto for the precise purpose of rating prediction by collaborative filtering [3]. It is thus highly appropriate for application on the ratings data in this project.

In essence, PMF hypothesises that the ratings matrix $R$ (rows are users and columns are books) is of low rank and tries to express this matrix as the product of a user and a book matrix. In other words, the underlying assumption is that the rating $R_{ij}$ for book $j$ by user $i$ can be expressed to some approximation by the dot product between a user descriptor (vector) $U_i$ and a book descriptor $V_j$. The length $D$ of both of these descriptor vectors (corresponding to the assumed rank of the ratings matrix) determines the accuracy with which the ratings can be reconstructed. Shorter descriptors (small $D$) may not be able to capture the underlying structure of the data, corresponding to the attitudes of users toward book aspects, and too long descriptors may explain the observed ratings very well but fail to generalize to unrated books (in machine learning this is called over-fitting), which is what we are really after.

The user and book descriptors are learned, i.e. fitted to the existing user-book-rating data. The way this works is that the entries of the $U$ and $V$ matrices, which are each just a collection of user and book descriptor vectors, respectively, are randomly initialized and iteratively updated to minimize a certain error function. This is done by gradient descent, i.e. following the gradient of the error function with respect to the entries of $U$ and $V$ to a minimum.

The error function being minimized by PMF is simply the squared difference between the predicted rating $U_iV_j$ and the true rating $R_{ij}$ from the training data. $U_i$ denotes the descriptor vector for user $i$ ($i$-th row of the $U$ matrix) and $V_j$ denotes the book descriptor for book $j$ ($j$-th column of the $V$ matrix).

$$\min \sum_{ij} I_{ij}(U_iV_j - R_{ij})^2 + \lambda(\sum_{id} U_{id}^2 + \sum_{dj} V_{dj}^2) \tag{2}$$

where the second term ensures that the entries of $U$ and $V$ do not become too large (too large entries lead to overfitting, i.e. matching the training data well but failing to generalize to new books). $\lambda$ is a parameter set by hand by trial and error that determines how stronly to enforce small entries. This method of encouraging generalization to new books is referred to in the machine learning literature as weight decay regularization.

### 3.3.2 Application

In order to obtain a rating prediction from the learned $U_i$ and $V_j$ vectors for a given user-book pair, the scalar product between the corresponding vectors is formed. However, in this project we are only indirectly interested in rating prediction—eventually we would like to recommend each user a few books that are believed to match her interest as well as a few other users that are believed to share some of that interest. The first task, book recommendation, is done by sorting all books according to the predicted ratings for a given user. The recommendations to each user are then the top-ranked books. We go into more detail on book recommendation in Section 4. Note that besides facilitating book recommendation, the PMF methods learns lower-dimensional descriptors for both users and books. Since they are derived from the ratings data, the user descriptors reflect to some degree the users' book interests. So we can use the user descriptors in $U$ to compare users—and recommend the most similar pairs to become friends. Section 5 discusses friends recommendation in detail.

# 4 Book Recommendation

**Abstract**

We discuss how we extend the method from the previous section to result in a personlized book recommendation engine. We present both postive quantitative and illustrative results by data visualization.

The goal of the book recommendation task is to make user-specific book recommendations based on the user's demonstrated book interests. In this section we discuss our methodology for generating book recommendations and how we evaluate them. We show the results of our best approach in terms of the evaluation metrics as well as more informal learned visualizations of the data. Finally, we discuss what could be done in the future as this project gets carried forward before presenting a brief summarizing discussion.

## 4.1 Methods and evaluation metrics

In Section 3.3.2 we discussed how book recommendation can be generated using a rating prediction method such as probabilistic matrix factorization (PMF), i.e. by simply recommending the books with the highest predicted ratings. This prediction-based approach is the core of our method. While a good evaluation metric for the rating prediction task itself would be the PMF error function of Equation 2, it does not fully express the goal of the book recommendation task, which is less concerned with correctly predicting low-rated books and places its emphasis instead on correctly retrieving *high*-rated books to recommend to the user. This task is not only found in collaborative filtering tasks, but also in other problems investigated by the ranking research community, such as query-based online document retrieval. A measure of success frequently used in this community is the Normalized Discounted Cumulative Gain (NDCG) [2].

$$NDCG(i, S_i)@n = N_i \sum_{j=1}^{n} \frac{2^{\text{rat(i, j)}} - 1}{\log(1 + k)} \tag{3}$$

Here $S_i$ are the scores (predicated ratings) assigned to all the books for user $i$; rat(i, j) is the $i$-th user's true rating of the $j$-th book after all the books are sorted according to $S_i$; and $N_i$ is the normalizing constant which ensures that the NDCG is always between 0 and 1. From Equation 3 we see that the NDCG scores the top $n$ prediction-rated books for a given user by their true rating in the ground truth data. Within those top $n$ books, the ground truth values of those with higher predicted ratings weigh exponentially more than the values of those with lower predicted ratings. Thus the NDCG function essentially says: the top $n$ predicted books for a given user have to have high ratings and the higher the ratings the better the NDCG score. In addition to serving as a sensible evaluation metric, the NDCG can be explicitly optimized by training a neural network to modify the user and book descriptors learned by PMF. Below we present the results from training PMF alone as well as further training to optimize NDCG.

The machine learning methods we use here are fairly computationally efficient. Training PMF on the condensed data (explained below) used for our experiements takes on the order of ten minutes on an up-to-date machine and scales linearly with the number of ratings in the dataset. NDCG training of a neural network on top of PMF training took an additional 30 minutes, but scales quadratically with the number of ratings.

## 4.2 Results

### 4.2.1 Quantitative results

Figure 3 compares the NDCG measures by histogram between the mean-rating baseline of Section 3.2, the PMF-based recommendations, and the PMF recommendations further optimized with respect to the NDCG by a neural network. The plots clearly show the merit of our methods in that they significantly raise the average NDCG over the baseline. The overall NCDG scores for each of the three methods are shown below (the higher the score, the better; see Equation 3).

| | Baseline | PMF training | PMF+NDCG training |
|---|---|---|---|
| NDCG score | 0.6560 | 0.7495 | 0.7517 |

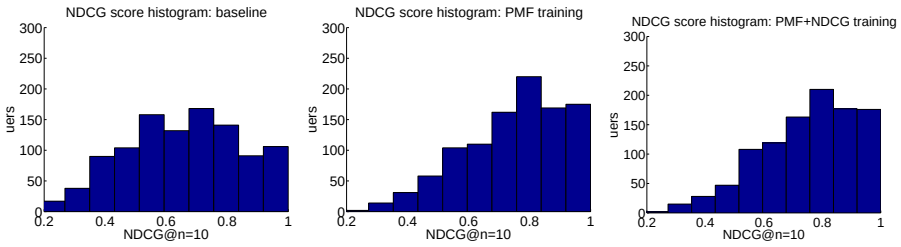


Figure 3: NDCG score histogram for baseline (left), PMF-training only (center), PMF+NDCG training (right)

Figure 3 is based on a condensed dataset where all users with very few ratings have been omitted. To quantify the effect the number of available ratings per user has on the quality of the results, we investigated the root mean squared error (RMSE) of the predicted ratings when training only the PMF model. The RMSE is simply the average difference between the true and predicted ratings on a test set of ratings held out during training. The table below shows the RMSE for three version of the dataset where more and more users have been eliminated for not having enough ratings. A trend towards a lower rating prediction error with more available ratings per user is clearly visible.

| | All users | $> 1$ rating | $> 5$ ratings |
|---|---|---|---|
| RMSE | 2.34 | 1.55 | 0.95 |

### 4.2.2 Learned data visualizations

Recall that PMF (even if followed by NDCG training) learns descriptors for users and books. Each book descriptor is a vector of length $D$. To visualize the relationship between books, these descriptors can be embedded into a two-dimensional space for display by dimensionality reduction methods like t-SNE [4]. t-SNE takes the higher-dimensional input representation (the descriptor vectors) and embeds each book into a two-dimensional space for plotting in a way that locally preserves distances. That is, two books with similar descriptors will appear close to each other in the plot. Figure 4 shows such a book embedding by displaying their titles in the t-SNE-learned locations. Only around 600 randomly chosen books are shown in order to keep the titles from overlapping too much to remain readable. Note that in several cases, books that are intuitively thought of as similar, actually appear close to each other, illustrating the value of the PMF-learned book descriptors.


Figure 4: Randomly chosen subset of book titles as learnt by PMF and laid out in 2-D by t-SNE

### 4.2.3 Variations tried

We started to explore the use of additional data besides the ratings data for training PMF rating prediction. We tried this by adding an extra term to the

PMF objective function in Equation 2 that resembles the error on the predicted ratings but instead tries to predict which books are on a user's virtual shelf. This initial attempt has not lead to a measurable improvement in rating predictions or book recommendations, so it will be left to future work to see if and how additional data sources can be included profitably.

### 4.3 Discussion and future work

In this section we discussed our methods for book recommendation and explained the used evaluation metrics. We showed that our methods extract sensible information for book recommendation from the data and demonstrated that denser ratings data (more ratings per user) results in higher-quality recommendations. Both the quantitative NDCG results as well as the more qualitative visualizations indicate the merit of performing book recommendation by PMF and NDCG training.

Furthermore, we have tried several ways of using the members' shelves data. In all instances we were never able to significantly outperform our current approach by using this extra information. The fact that shelves only provide us with one-sided data is certainly one challenge. Another is that ratings data is more expressive and as such encodes more information.

# 5 Friends Recommendation

**Abstract**

We discuss how to augment the rating prediction method of Section 3 to result in a friends recommender as well as how we evaluate the resulting system. We demonstrate the effectiveness of our approach both qualitatively and by data visualization. It is furthermore shown that the existing Indigo community friendships are somewhat correlated with users' book interests.

The goal of the friends recommendation task is to find pairs of users with maximally similar book interests. The matched-up users can then be recommended to become social networking friends in Indigo's online book lovers community. In this section, we discuss our approach both in recommending friends as well as in evaluating the quality of the used method. We present our results and an illustrative data visualization as well as discussing potential future work on this topic.

## 5.1 Approach

As opposed to the commonly studied book recommendation problem, to our knowledge, friends recommendation is actually a new task and no common approach exists for either the task itself nor its evaluation. In this subsection, we will discuss how we approach the friends recommendation task as well as convince ourselves of the sensibility of the results.

### 5.1.1 Recommending friends

Our friends recommendation approach is based on the same underlying machine learning methods as book recommendation: probabilistic matrix factorization (PMF) with potential subsequent normalized discounted cumulative gain (NDCG) training of a neural network. Recall that both methods yield a descriptor vector for each user and one for each book. For the purpose of friends recommendation, we only consider the user descriptors (which contain the users' book preference information). Each user descriptor vector can be though of as a point in a $D$-dimensional space, so we can compute (Euclidean) distances between the points representing users in that space. If the training on the ratings data successfully extracted users' book interests, then users with similar interests will have similar descriptor vectors with small distances in the vector space. In order to make actual friend recommendations for a given user, we sort all other users in the database by their distance to this user in the descriptor space and recommend the top $k$ users as potential friends.

### 5.1.2 Evaluating recommendations

How to best evaluate friends recommendations depends strongly on the desired properties of friend pairs. For the purposes of this project, the client suggested

that friends should share common book interests as this would most likely increase the liveliness of the online book lovers community. "Common book interests" is a rather vague term and is not measurable quantitatively. We thus rely in this project on suggestive evidence by a quantitative measure *normalized overlap* that intuitively captures the idea "common book interest". This and similar measures are also used, for example, in the natural language processing research community for comparison of document topics.

The normalized overlap is defined on binary vectors (containing only 1's and 0's) as follows. Let $v$ and $w$ be two binary vectors of the same vector space. Then the normalized overlap is

$$O(v, w) = \frac{\sum_i [v_i == w_i]}{\min(\sum_i v_i, \sum_i w_i)} \tag{4}$$

$\sum_i [v_i == w_i]$ counts the number of positions that contain a 1 in both vectors, while $\sum_i v_i$ and $\sum_i w_i$ count the number of non-zero entries in each of the two vectors separately. So the normalized overlap between two binary vectors is simply the proportion of shared non-zero entries compared to the vector with fewer non-zero entries. A normalized overlap of 0 indicates no common entries whereas a value of 1 indicates a perfect match.

We use the concept of normalized overlap to evaluate our friends recommendations as follows. For each user, we sort all other users by their learned descriptor distances to the query user and keep the top (closest) $k$. For each pair of a user and one of its top $k$ recommended friends, the normalized overlap between their high-ratings vectors is computed. A user's high ratings vector is simply a binary vector containing a 1 for each book that the user has rated 4 or higher and a 0 for all other books (either rated low or not rated at all). Similar overlap scores can be computed based on the shelves or purchase history as well as ground truth friends data, which are all already binary.

## 5.2 Results

### 5.2.1 Friends recommendation

Here we present the overlap evaluations of our friends recommendation results. Most results are presented for two versions of the dataset: one containing all users and books with at least one rating, and one with only users who have rated at least 16 books. This latter somewhat denser dataset contains 863 users and 2531 books for a density of 0.0037 (compare to full dataset statistics in Section 2.1.1).

Figure 5 shows the averaged normalized high ratings overlap for both versions of the dataset. For each user, the normalized overlap was computed for the top 100 suggested friends and these values averaged over all users. The figure shows a clear maximum of overlap for high-ranked suggested friends, with the overlap falling off for lower-ranked suggested friends on average. This is exactly what we want—suggested friends should maximally agree in their book interests—illustrating that PMF learns sensible user descriptors.

Figures 6 and 7 show similar plots for virtual bookshelves and purchase history overlap, respectively, again confirming the suitability of friends recommendation based on PMF-learned user descriptor similarities. The condensed data plot for purchase history overlap is missing as the mapping between different book IDs in the dataset could not be resolved within the time frame of the project. Note also that the numerical values for purchase history overlap are significantly lower than for high ratings and bookshelves overlap. One possible reason for this is that people may buy books not only for themselves, but often as gifts for others with whom they might not share book interest.

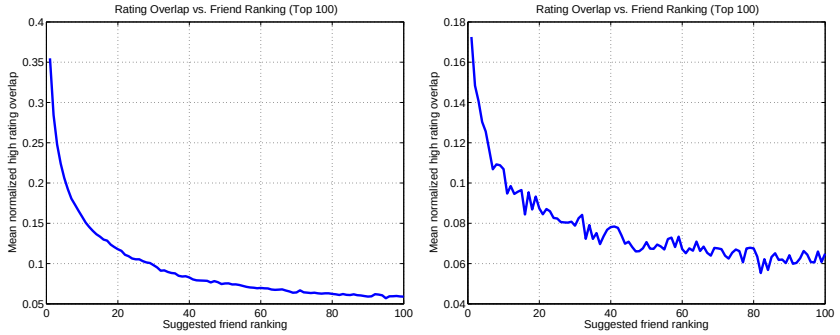


Figure 5: Average normalized high ratings overlap based on PMF training on the full data (left) and the condensed data (right)

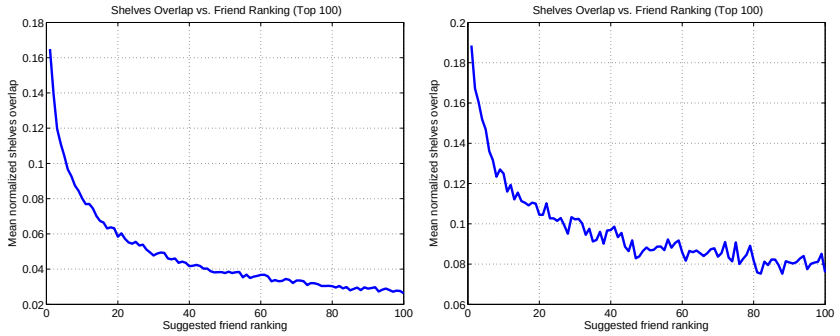


Figure 6: Average normalized virtual bookshelves overlap based on PMF training on the full data (center) and the condensed data (right)

### 5.2.2 Current community friends data

Recall that the reason that we do not use the existing community friends data to validate our methods was the client's hypothesis that existing community

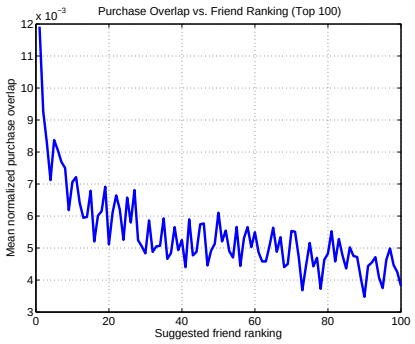


Figure 7: Average normalized purchase history overlap based on PMF training on the full data

friendships are based on real world friendships as opposed to common book interest. Here we test this hypothesis in two ways. First we attempt to induce new friends recommendations based on only the existing friends data by spectral analysis and evaluate the resulting friends recommendations in terms of common book interests. Secondly, we use the PMF-induced friends ranking to compare the social network (in terms of existing friends) of pairs of users that we recommend to be friends.

Spectral analysis is a common machine learning technique used to generate a data representation that clusters better than the original representation. A good tutorial on spectral clustering can be found in [5]. We performed spectral analysis on the existing friends data from Indigo's online community, resulting in a set of user descriptors that can be used to include potential friends rankings in the same way as the user descriptors learned by PMF. These friend recommendation rankings will include as high-ranked suggested friends the already existing friends of a user as well as those sharing common friends. Based on these friend recommendation rankings, Figure 8 shows the high rating overlap for top 100 ranking potential friends per user averaged over all users. Note that a noisy but clear downward trend in high rating overlap (i.e. common book interest) can be observed as we move further away from the top-ranked recommended friends. Since the underlying suggested friends ranking is based solely on the existing friends data, this means that the existing community friendships do in fact to some degree show common book interests.

Figure 9 shows the average normalized overlap of ground truth friends from the existing friends data for suggested friends ranking based on PMF-training on the book ratings data. Note that while noisy, the figure shows a clearly rising existing friends overlap as we move to higher and higher-ranked suggested friends. This means that on average, pairs of users that are suggested to be friends by our methods based merely on their common book interests as expressed in their book ratings, are more likely to have common third friends

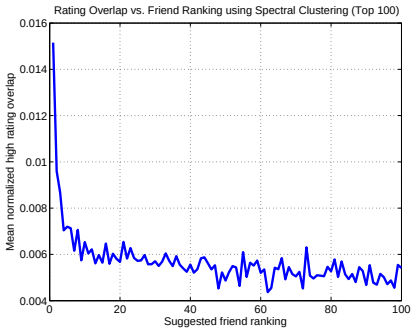


Figure 8: Average normalized high ratings overlap based on spectral analysis of friends data

than pairs of users with less similar book interests. This result again suggests that existing community friendships do correlate with common book interests, to some degree invalidating the hypothesis that existing community friendships do not at all reflect book interests.

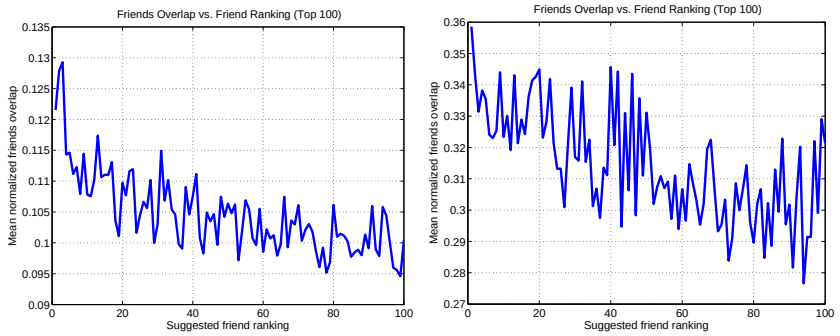


Figure 9: Average normalized ground truth friends overlap based on PMF training on the full data (left) and the condensed data (right)

### 5.2.3 Variations tried

We started to explore the use of additional data besides the ratings data for learning user descriptors by PMF. One way of doing this would be to add an extra term to the PMF objective function in Equation 2 that resembles the error on the predicted ratings but instead tries to predict which books are on a user's virtual shelf or in the purchase history. We tried this with the virtual shelves data, but that did not result in an improvement in friends recommendations as measured by similar overlap plots as the ones presented above. Future work

could be invested to see whether improvements using the shelves or purchase history data are possible.

#### 5.2.4 Learned data visualizations

To visualize what was learned by PMF on the users, we reduced the dimensionality of the user descriptors to two dimensions using t-SNE for display [4]. In the top plot of Figure 10, each user is shown as a number corresponding to the most frequently occurring super category on the user's bookshelf. Red numbers correspond to the most frequently occurring super category for a given user and blue ones to the second most frequently occurring super category per user. The size of each number corresponds to the proportion of books on the user's shelf coming from that super category. Numbers are displayed close to each other if the corresponding users have similar PMF-learned descriptors. Thus several same numbers in a cluster indicate the method successfully discovering sets of users with common book interests. Note that the map is noisy for several reasons. Neither PMF-training nor dimensionality reduction with t-SNE are perfect, and only the top two super categories per user are displayed, giving a sense of the user's book interests but not fully defining them.

The heat maps at the bottom of Figure 10 show the same map as the top, except each heat map shows only user concentrations for a single super category per map. Warmer colors indicate a higher concentration of users with this super group's books on their shelves. Note that for each of the shown super categories, the concentration of users for whom this category is the most frequently occurring one on their book shelves is fairly localized on the map. This again shows that PMF succeeds in learning a similarity measure between users (on which the maps are based) which correlates with users' book interests.

### 5.3 Discussion and future work

In this section we discussed our methods for generating as well as evaluating friends recommendations. We showed that our methods learn sensible user representations according to their book interests that can be used to successfully recommend pairs of users for friendship. Two-dimensional visualizations were given that further illustrate our methods' effectiveness.

We have also briefly experimented with including the users' virtual book shelf data, but these initial attempts have been unsuccessful in increasing the quality of friends recommendations. Future work is necessary to fully determine if and how such additional data can be used profitably.

| | |
|---|---|
| 1 - | Travel |
| 2 - | Sports |
| 3 - | Entertainment & Leisure |
| 4 - | Business & Finance |
| 5 - | Hobbies |
| 6 - | Law & Order |
| 7 - | Fiction |
| 8 - | History |
| 9 - | Self |
| 10 - | Special Interest |
| 11 - | Health & Living |
| 12 - | General Interest |
| 13 - | Other |
| 14 - | Science |
| 15 - | Family |
| 16 - | Children |
| 17 - | Arts |
| 18 - | Animas |
| 19 - | Religion |
| 20 - | Music |

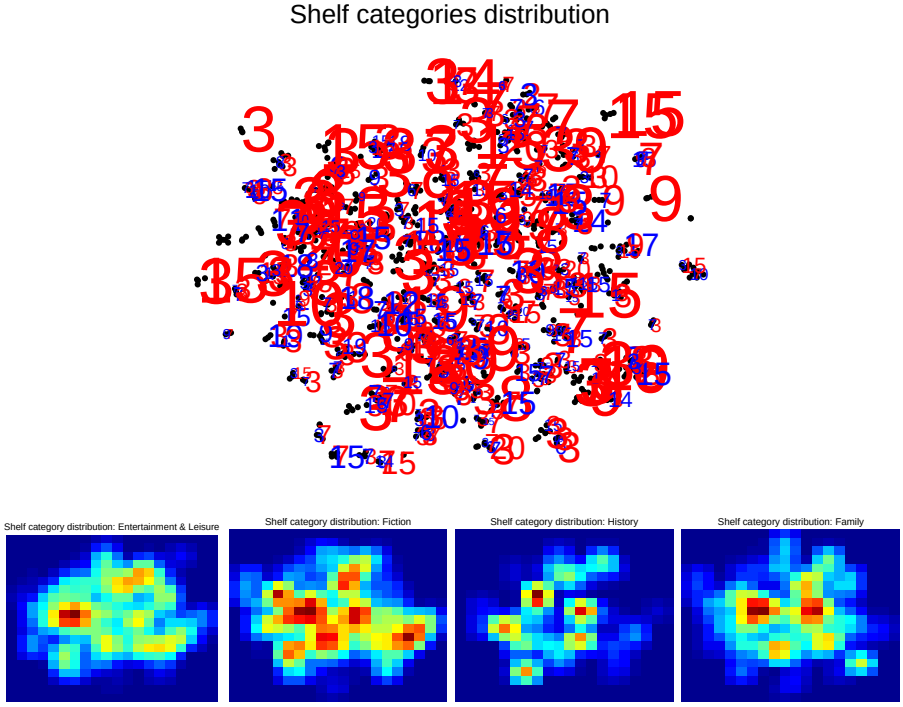


Figure 10: Concentration of users with similar books interests (see text for details)

# 6 Discussion

**Abstract**

Here we conclude the report by summarizing our achievements and results as well as the challenges that are inherent in the task and the data. We propose an initial working solution to the original problem and give a few other recommendations that are based on our experience working with this data.

## 6.1 Summary of achievements

In this report we have presented the results of our project work on recommending books and friends based on users' book interests. We have applied and adapted a unified method for generating both book and friends recommendation, enabling us to do both while having to train only one significant machine learning algorithm. The book recommendations generated by our method are significantly better and more personalized than the baseline. We formulated a measure to evaluate friends recommendations and showed that our method for recommending friends is successful by that measure. We furthermore presented insightful data visualizations for both the book and friends recommendation tasks to further illustrate our method's effectiveness.

## 6.2 Summary of challenges and future work

The primary challenge in extracting user book interests from data, and thus in recommending them books or friends, remains the sparsity of the available data. It may thus be profitable to concentrate any future work on the following two ways of improving results. First, more data, e.g. more ratings per user, should be acquired if possible. Secondly, extending our methods to make use of more data sources simultaneously may prove especially important in the presence of very sparse data.

## 6.3 Solution proposal

Given our current insights and results, if asked to implement a book and friends recommendation engine for live employment in Indigo's online operations, here we describe what we would do.

First, the system should be implemented according to the methods we describe in this report. That is, PMF should be trained on ratings for community users to produce the book and user descriptors used for both recommendation tasks. Books will be recommended by a predicted rating-induced ranking and friends will be recommended according to the similarity of user descriptors. Live test should then be run over a sufficiently long period to determine the effect of these recommendations on purchase behavior and community interaction. Apart from the actual monetary value of providing recommendations, other factors should be carefully measured. For example, the effectiveness of

recommendations based on user profiles should be examined. It is very likely that users will only benefit from recommendation once they have a certain number of ratings in the system. Until they do, it might be more appropriate to recommend books based on purchase history to such users (i.e., "Customers who bought this item also bought").

## 6.4 Recommendations to the client

Even in the absence of manpower to implement the book and friends recommendation solution described in this report, some less costly actions could be taken right away to maximize the long-term value of the continuously acquired and logged online member data. We already mentioned that more ratings per user would drastically increase the recommendation quality. Likewise, a higher diversity of ratings would provide more material for the machine learning methods to learn from. For example, if online users could be encouraged to rate the books they have bought, more negative ratings could be acquired. This should more accurately define users' book interests by retrieving a more balanced distribution. Currently users rate only at their own impulse, heavily skewing the distribution of ratings toward the positive end. Also for the purpose of increasing the amount of available ratings data, ratings from non-community users, if they are logged in, should be tracked. It seems that currently, book ratings by non-community users are recorded without a user identifier. Even if book recommendations are not to be done to those users, the additional rating data will help the collaborative filtering method make better recommendations to community users through the "wisdom of the crowd" effect.

# References


[1] ChoiceStream Inc. 2008 ChoiceStream Personalization Survey, 2008.

[2] Kalervo Jarvelin and Jaana Kekalainen. IR Evaluation Methods for Retrieving Highly Relevant Documents. In *SIGIR*, 2000.

[3] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic Matrix Factorization. In *Advances in Neural Information Processing Systems*, volume 20, 2008.

[4] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, November 2008.

[5] Ulrike von Luxburg. A Tutorial on Spectral Clustering. Technical Report TR-149, Max Planck Institut für Biologische Kybernetik, 2006.