Comparative Analysis of Collaborative Filtering-Based Predictors of Scores in Surveys of a Large Company

Markos F. B. G. Oliveira[1,2], **Myriam Delgado**[1], **Ricardo Lüders**[1]

[1]CPGEI / DAINF – Universidade Tecnológica Federal do Paraná (UTFPR)
Av. Sete de Setembro 3165 – 80230-901 – Curitiba – PR – Brazil

[2]Pin People Tecnologia Aplicada a Pessoas Ltda.
Cubo Itaú, Rua Alameda Vicente Pinzon 54 – 04547-130 – São Paulo – SP – Brazil

markos.flavio@pinpeople.com.br, {myriamdelg,luders}@utfpr.edu.br

Abstract. *Collaborative Filtering (CF) can be understood as the process of predicting the preferences of users and deriving useful patterns by studying their activities. In the survey context, it can be used to predict answers to questions as combinations of other available answers. In this paper, we aim to test five CF-based algorithms (item-item, iterative matrix factorization, neural collaborative filtering, logistic matrix factorization, and an ensemble of them) to estimate scores in four survey applications (checkpoints) composed of 700,000 employee's ratings. These data have been collected from 2019 to 2020 by a large Brazilian tech company with more than 10,000 employees. The results show that collaborative filtering approaches provide relevant alternatives to score questions of surveys. They provided good quality estimates. This result can be further explored to eventually reduce the size of questionnaires, avoiding burden phenomena faced by respondents when dealing with large surveys.*

1. Introduction

Startups dedicated to collect and analyze data from employees in order to provide a better understanding of the work environment, the so-called Human Resource Techs (HR Techs for short), are gaining visibility in the market. Periodically assessing employees' perceptions and sentiments about several aspects of the organization is helping companies to grasp people behaviors and (hidden) patterns inside the organizations. Furthermore, this data-driven process allows human resource departments to make decisions quickly and more reliably.

However, one of the most systematic ways to obtain information on people inside companies is still the survey application. A typical survey methodology is to periodically (e.g., every two quarters) apply an organizational climate survey that tries to capture employee's perceptions across a broad range of aspects from mental health to teamwork. These surveys tend to be large (up to 300 questions) and frozen (in terms of content, sequence, and amount of questions) for all respondents. Despite the advantage of this kind of methodology, especially regarding design, application, and sampling control, a surveying process based on immovable questions has several issues. It is known, for instance, that there is an inversely proportional relationship between questionnaire size and response rate. It occurs because respondents are more likely to not opening extensive questionnaires. If they do open it, they are more likely

to leave the questionnaire early (*survey breakoff*) [Early et al. 2017, Zhang et al. 2020]. Moreover, respondents may give answers to finish the questionnaire more quickly to reduce their cognitive effort. They either select arbitrary answers (such as the maximum score for all answers) or choose answers that allow them to shorten the questionnaire ("no" answers, for example). This behavior is called *satisficing* [Lavrakas 2008], which is more frequent in large questionnaires occurring especially in the last questions [Early et al. 2017, Zhang et al. 2020, Gonzalez and Eltinge 2008].

In short, companies can lose valuable data due to survey breakoff or lose data quality due satisficing since large questionnaires are more susceptible to respondent burden. To overcome those issues, some works suggest adaptive survey frameworks that reduce or reorder survey questions. For example, in [Zhang et al. 2020] the authors propose a reduction strategy based on active learning by selecting questions that maximize the precision of a Matrix Factorization (MF) model. This model can be further used to recover the scores that were missing due breakoff or reduction itself. [Boim et al. 2012] approach takes the reduction problem as an optimization one, trying to minimize the uncertainty present in systems due to the different distribution of answers of each question.

In the present work, as an attempt to learn the answering behavior, we focus on building predictive models using collaborative filtering techniques. It is a first step for building recommender systems, since accurate estimates are necessary to guide the further recommendation step. We show that it is indeed possible to learn these estimates given enough data, which could be used as a substrate of a recommending system capable of designing smaller questionnaires toward well-defined objectives. In this way, we apply five algorithms, four based on collaborative filtering (CF) and one ensemble approach, using more than 700,000 employee ratings. These data were collected from four surveys applied between 2019 and 2020 from a large Brazilian tech company with more than 10,000 employees. Results show that the scores given by employees have patterns that can be explored to reduce the questionnaire eventually. The main contribution of the present paper is applying collaborative filtering algorithms to surveys for estimating scores of questionnaires answers. It is an expansion of the application frontier of this technique, once there is no literature linking CF as a tool for assessing question scores in surveys, particularly considering Likert scale responses.

2. Related works

Most surveys are often large and fixed in terms of sequence and number of questions for all attendants because HR departments of companies wish to know as many aspects as possible about the organization. Even though this kind of survey is simple to apply, it suffers from user burden phenomenon. Such a phenomenon negatively impacts the respondent's experience of answering the survey and the posterior quantitative analysis on the data because a poor experience leads to bad quality responses. To tackle these issues, some works develop adaptive or dynamic questionnaire design strategies.

Adaptive survey design (ADS) found in [Chun et al. 2018, Schouten et al. 2013, Wagner 2008] is a related study field that uses data-driven information to build high-quality surveys. However, ADS has a more generic appealing as it considers several aspects of the survey to engage more participants to attend and finish them. For example, ADS may study the expected decrease in the nonresponse rate (the proportion of nonre-

spondents with respect to those who were contacted) with a change in the contact mode employed (telephone, e-mail, etc.).

A different line of research tries to select questions that reduce uncertainty over the answers. [Early et al. 2017] chooses questions that maximize the information gain measured by conditional entropy in a cost-sensitive strategy. In that work, questions that require more cognitive effort are given more penalty and have less probability of being sampled. The same idea is followed by [Boim et al. 2012], but it approaches the problem using a constraint optimization formulation with no cost penalties.

Some works treat the problem of generating reduced questionnaires through an active learning perspective. Active learning is a learning approach that tries to select the most informative data for a prediction model. In this context, [Zhang et al. 2020] proposes a reduced questionnaire engine that selects questions to the user based on their capacity to maximize the precision of an MF learning model. This model can then reconstruct the user-question table, filling the empty entries with accurate estimates. Similarly, [Early et al. 2017] aims to maximize survey completion by selecting questions that most reduces the prediction uncertainty of an auxiliary model that predicts a variable of interest. In this case, if a breakoff occurs, the model has maximized its learning from the submitted answers. Finally, in [Ortigosa et al. 2010], a decision tree is built with questions as nodes. The tree classifies learning styles from students' answers, a task that originally demands a large number of questions to be made. By using this model to guide the sequence of questions, one can only ask the relevant questions for the prediction problem of interest.

Perhaps the most similar work to ours is found in [Zhang et al. 2020]. Unlike our work that is meant to be used during the questionnaire to estimate unknown answers, the model proposed in that work is useful to understand hidden relationships on data. For instance, it is possible to identify the most informative questions.

The present paper focuses on training collaborative filtering models that try to predict employees' answers from a set of Likert scale questions, each one evaluating a particular aspect of the company. It differs from model-oriented approaches (that use active learning) that try to classify some other quantity from the obtained answers rather than the answers themselves. In these works, questions are selected to make accurate the predictions on that quantity, even with less information.

3. Problem Description

The most common way of large organizations assessing employees' perceptions is from survey application. The goal of these surveys is to capture employees' feelings and review about a broad set of aspects of the company. Although there are several survey methodologies and a heterogeneous set of question types ([Krosnick 2018]), HR departments usually focus on surveys with Likert scale questions. In this type of question, the audience receives a range of options (often 5 or 7) to measure how favorable each respondent is about a particular aspect. In this setup, respondents are called "favorable" to the aspect being assessed when they answer 4 or 5 in a scale of 1-5, or 6 or 7 in a scale of 1-7.

The literature of recommender systems is vast, and several algorithms have been proposed to estimate an unknown *event* r_{ij} between a user i and an item j. There are several possible events that r_{ij} may model, such as click events, number of purchases,

or explicit ratings given by users to items in the past. Therefore, the signal type of r_{ij} depends on the type of information available to the system. If the system has access to explicit ratings of users on items, then the interaction captured and generalized by the system is how people give ratings to items in this scale. For instance, $r_{ij} \in \{1, 2, 3, 4, 5\}$ if ratings are given in a scale of 1-5 stars. If only implicit information is available, such as item purchases or views, $r_{ij} \in \{0, 1\}$ and estimates represent the likelihood of user interacting with an item. The implicit interaction embeds less information because a zero ("0") value does not mean a user doesn't like an item. In this case, she may not know the item exists. However, implicit data is easier to get being the usual signal available to recommender systems because most people do not explicitly rate items [Johnson 2014].

In this work, we use employees' answers from questionnaires to build a prediction model using collaborative filtering algorithms. Each question maps to a numeric answer that explicitly measures the favorability of the employee to an aspect of the company. By knowing estimates of answers to the questions that were not asked, one could reduce the questionnaire size by focusing on questions that are *relevant*, e.g., the ones that would be given low scores.

The problem of estimating events can be stated as follows. Consider a partially filled matrix $R_{N \times M}$ of N users (respondents) and M items (questions). Each known entry r_{ij} of R, with $i = 1 \ldots N$ and $j = 1 \ldots M$, is a rating that measures some kind of positive perspective user i took from a past experience with item j. More pleasant experiences have higher ratings. In general, r_{ij} is taken from a fixed, small and ordinal set of values such as $\{1, 2, 3, 4, 5\}$ or $\{0, 1\}$. In most problems, R is sparse (with missing data) because most users do not rate most items. We consider the set Ψ_i of items rated by user i, the set Ω_j of users who gave rates to the item j, and the set Ω of user-item pairs whose ratings r_{ij} are known. The problem is building a predictor model that estimates unknown ratings of R (unanswered questions in our problem). A proxy $\hat{R}$ is built from R so that an error metric between R and $\hat{R}$ is minimized. Elements of $\hat{R}$ are denoted by $\hat{r}_{ij}$ and are used for guiding the recommendation engine.

When a new questionnaire is released, no answers about users and questions are known, i.e., the matrix R is completely empty, and no recommendations are possible. Two conditions are necessary for the system to start making meaningful recommendations for a user u: (i) a substantial amount of people has submitted their answers before u has started to answer the questions, and; (ii) u has already answered a few questions.

We foresee three solutions to this problem. The first solution is to apply a content-based approach, such as the one proposed by [Al-Shamri 2016]. It uses respondents' demographics to generate recommendations. This approach dismisses the second condition above, being possible to generate recommendations for u from the first question of the questionnaire. A second solution is to use active learning and exploration techniques to guide the recommendations for the first answers. For example, the proposal of [Zhang et al. 2020] could be used to guide the recommendation engine with the goal of maximizing the performance of an MF prediction model. The engine could switch to a second recommender that uses the prediction of the model being learned if the performance of such model achieves a certain level. Other algorithms seeking optimal exploration-exploitation trade-offs (with and without contexts) can be used in this sense [Li et al. 2016, Wang et al. 2017, Wu et al. 2016, Song et al. 2014]. The third possible

solution is to use the history of answers from old questionnaires. [Koren 2009] presents an approach that embeds meaningful former events by including additional biases in the MF formulation.

4. The Addressed Predictor Models

There are two main steps for making recommendations. The first one is to estimate unknown interactions between users and items. The second step is to rank the items based on estimates. This step is more qualitatively involved and depends on the problem and business interests. A common approach for explicit problems is to sort items in descending order of estimates.

The literature on recommender systems focuses on the first step. Then, explicit or implicit benchmark data sets such as MovieLens[1] are used to assess the relative performance (usually RMSE) of approaches against each other. Predictor models receive as input a built-in matrix R that, even know is sparse, it contains millions of known user-item events. From R, the algorithms build models capable of generating predictions ($\hat{R}$) that ultimately can be used for recommendations.

These algorithms are often categorized into two broad classes: collaborative filtering (CF) and content-based algorithms. In CF, every past interaction (known values from R) collaborates with the system in understanding how user u interacts with item i. More explicitly, all ratings $r \in R$ are used to evaluate $\hat{r}_{i,j}$; not only those related with u or i. These systems have great performances in benchmark data sets [Su and Khoshgoftaar 2009], being the most common choice. As a consequence, most of the current State-of-the-Art of RS is built on CF systems or hybrid systems that combine CF with other techniques. Thus, CF is the methodology we approach in this paper.

CF differs from content-based approaches, in which each item (or user) is mapped to a profile of known features values [Pazzani and Billsus 2007]. For example, a question can be mapped (by a specialist) to a vector of values, each value measuring the degree of intersection between the question and semantic concepts, such as leadership, management, etc. A separated model M_u is learned for each user u from a set of examples (old answers) in a supervised framework. Inputs are the feature vectors of items rated by u while the outputs are the ratings. Only the past behavior of user u on items $i \in I_u$ is used for predictions; i.e., there's no collaboration. Content-based approaches are less appealing than CF mainly because 1- it is difficult to access an appropriate set of features for items (which is even more critical for a growing set of items), and 2- the system is overspecialized, never recommending items outside user's tastes.

Hybrid approaches that combine CF and content-based are also possible. [Melville et al. 2002] first uses a content-based approach to estimate the unknown values of R from vector abstractions of each item. Then, they feed this pseudo-ratings matrix to a collaborative algorithm that refines the estimates.

CF algorithms can be categorized into memory-based or model-based [Su and Khoshgoftaar 2009]. Memory-based CF algorithms explicitly rely on the correlations of the ratings between the most similar users or items to generate predictions for user-item pairs. In contrast, model-based algorithms try to approximate R with $\hat{R}$ so that

[1] https://grouplens.org/datasets/movielens/

an error between the R and $\hat{R}$ is minimized. $\hat{R}$ is factorized into a user and item matrices that have reduced dimensionality k. These matrices are multiplied to build $\hat{R}$. Each user u and item i are described with k latent factors that are learned iteratively from the known ratings of R.

We have selected one memory-based algorithm (item-item), three matrix (model-based) factorization algorithms and one ensemble approach encompassing all of them. Each one is representing a relevant approach of the literature for recommender systems using collaborative filtering [Rendle et al. 2020, Kulkarni et al. 2020]. The following sections explain the particularities of each chosen approach.

4.1. Item-item Collaborative Filtering

Item-item CF is a memory-based model that relies on the similarity of item ratings to guide recommendations. An item is recommended for a user if the user has liked similar items in the past. Two items are similar if users have given similar ratings to them. Similarity is often measured by cosine similarity or Pearson correlation. [Al-Shamri 2016] and [Su and Khoshgoftaar 2009] describe several other similarity measures that can be used to compare items or user profiles. Although its conceptual simplicity, it is a very competitive approach as it provides similar performance results compared to matrix factorization predictors. Formally, the estimate $\hat{r}_{ij}$ is given by (1),

$$\hat{r}_{ij} = \bar{r}_j + \frac{\sum_{j' \in \Psi_i} w_{jj'}(r_{ij'} - \bar{r}_{j'})}{\sum_{j' \in \Psi_i} |w_{jj'}|} \tag{1}$$

with $w_{jj'}$ being the similarity measure between items j and j', Ψ_i being the set of items rated by user i, and $\bar{r}_j$ being the average rating of item j (same for $\bar{r}_{j'}$). Notice that the score given by this method is the average rating of the item plus a weighted average of rating deviations instead of pure ratings. In this case, each user has its own bias of rating items: a conservative user may consider only five (5) as a good score, while a more flexible user may consider three (3) as good enough. Thus, it measures how much user i likes j' compared to how much the population of users likes j'. The similarity measure can be given by the Pearson-correlation coefficient between rating distributions of j and j' provided by all users who rated both j and j'.

User-user CF works similarly to item-item, but it recommends items for users based on user similarities instead of item similarities. Nevertheless, item-item CF is often preferred because its weights are more accurate. That is the case because, in most problems, two items have a lot more users in common than two users have items in common. Item-item is also computationally faster for problems where $N >> M$, which is the general case.

4.2. Iterative Matrix Factorization

In the matrix factorization algorithm, the goal is to build $\hat{R}$ as a low-rank approximation of R. $\hat{R}$ is then factorized into two matrices, as shown in (2).

$$\hat{R}_{N \times M} = W_{N \times k} U^T_{M \times k} \tag{2}$$

In the user matrix W, each row represents a vector of k latent features of a user. Similarly, the item matrix U has rows that hold items representations. Users and items

are then projected into a latent shared space with dimensionality k. This factorization approach is similar to a truncated SVD (Singular Value Decomposition) that approximates a matrix A to $U_{N\times k}S_{k\times k}V^T_{M\times k}$ if A is full rank and $N > M$. By substituting $W = US$, we get the same approximation of (2).

One possible way of building $\hat{R}$ is to iteratively update the vectors representations of W and U towards the minimization of the mean square error between R and $\hat{R}$. To assess the individual error between a prediction $\hat{r}_{ij}$ and the true rating r_{ij}, the regularized loss function of (3) could be used,

$$J = \sum_{i,j\in\Omega} (r_{ij} - \hat{r}_{ij})^2 + \lambda(||W||^2_F + ||U||^2_F + b_2^2 + c_2^2) \tag{3}$$

with L2-regularization penalty λ, Frobenius norm $||\cdot||^2_F$ and $\hat{r}_{ij}$ built according to (4),

$$\hat{r}_{ij} = w_i^T u_j + b_i + c_j + \mu \tag{4}$$

with the user vector w_i, the item vector u_j, the bias for user i b_i, the bias for item j c_j and μ as the average rating of R. Equation (4) is the most common way of designing the prediction rating for matrix factorization models. The biases are included to model existing phenomena related to user-item ratings, such as user optimism or item popularity, and usually increase performance [He et al. 2017]. In [Koren 2009], the use of these biases was developed to capture temporal dynamic effects, such as users' perception changes about items. Alternating least squares (ALS) can be used to find the model parameters that minimize J from the closed-form solutions built from its derivative.

4.3. Neural Collaborative Filtering

The neural collaborative filtering (NCF) from [He et al. 2017] is an appealing approach for implicit data sets that has been gained attention in recommender systems community due to its high scalability and flexible network architecture.

In NCF, the dot product between w_i and u_j is replaced by a more complex function that is learned by backpropagation. In its simplest form (used in this work), the network receives the interaction vectors $row_i(R)$ of user i, and $col_j(R)$ of item j. Then, it generates latent vectors from both inputs and concatenates them into a single vector. This vector is passed to a sequence of fully connected layers. The activation function of the output unit is a logistic function suitable for implicit data providing $\hat{r}_{ij} \in [0, 1]$.

The network is trained to minimize the binary cross-entropy loss function of (5). J is then minimized with stochastic gradient descent (SGD).

$$J = -\sum_{i,j\in\Omega} r_{ij} \log \hat{r}_{ij} + (1 - r_{ij}) \log (1 - \hat{r}_{ij}) \tag{5}$$

An advantage of this model is that it can generate predictions for new users and items directly from their ratings without retraining. It is not the case for matrix factorization models (and item-item) as their embed representations that generate predictions are learned during training. New training cycles that include new users (or items) are required, or handcraft rules to handle these cases must be created. Making recommendations with new users or items is referred to as the cold-start problem [Lika et al. 2014].

4.4. Logistic Matrix Factorization algorithm

The logistic matrix factorization is a CF algorithm designed for implicit data problems [Johnson 2014]. Similarly to NCF, it models the probability of a user preferring an item by using a logistic function but restricting $\hat{r}_{ij}$ to a linear function, as shown in (6).

$$\hat{r}_{ij} = w_i^T u_j + b_i + c_j \tag{6}$$

Equation (7) shows the log-likelihood function of parameters w_i, u_j, b_i, and c_j.

$$L = \log[p(W, U, b, c \mid R)] = \sum_{i,j \in \Omega} \alpha r_{ij} \hat{r}_{ij} - (1 + \alpha r_{ij}) \log(1 + \exp(\hat{r}_{ij})) - \frac{\lambda}{2} w_{i2}^2 - \frac{\lambda}{2} u_{j2}^2 \tag{7}$$

with α being a hyperparameter that weights observations and λ being the usual L2-regularization penalty (applied to the latent vectors only). The gradient descent method is used for maximizing L. For training time reduction, [Johnson 2014] suggest using a negative sampling procedure and an adaptive learning schedule with AdaGrad.

5. Experiments and Results

This section considers the application of the algorithms described in Section 4 to predict employees' answers that are not favorable to company aspects. Algorithms consider data from four different questionnaires (called *checkpoints* as shown in Table 1) applied from 2019 to 2020 by a large technology Brazilian company. Those questionnaires were chosen since they refer to the same periodic climate survey, providing a great intersection of questions and respondents between checkpoints.

The original answers are given in 1-5 Likert scores. In this scale, respondents that answer 1, 2, or 3 to a particular question are called "not favorable" to that aspect. To identify this event, we approach the prediction problem as a binary classification problem whose positive class (the one we want to identify) relates to answers 1, 2, and 3, while the non-positive class relates to answers 4 and 5. It is still explicit data because the class zero ("0") is explicitly given by the employee - with the highest answers. That's the first step for building a reduced questionnaire that seeks to identify and suggest questions about aspects that employees negatively view. Moreover, users that do not answer the most common questions (i.e. the questions with less than 1% of breakoff rate) are removed. This filtering process is applied to eliminate missing values, and it is enough to build a user-question table with complete information. Table 1 presents filtered data information with the questionnaires identified by a checkpoint ID given by cp_t.

Table 1. Checkpoints covered in this work. Consecutive checkpoints represent consecutive surveys periodically applied from 2019 to 2020.

Checkpoint	# Respondents	# Questions	# Ratings
cp_1	3512	25	87,800
cp_2	5491	31	170,221
cp_3	5299	41	217,259
cp_4	5343	51	272,493

An exploratory analysis of data reveals that people are more likely to give higher scores than lower ones. This behavior occurs for the whole survey counts and for individual questions as well, with few exceptions. This makes the prediction problem imbalanced, as only 20% of the answers are 1, 2, or 3.

As the set and number of users and questions for different checkpoints vary (although there are intersections), we decided to evaluate the different checkpoints separately. Therefore, each checkpoint is considered a completely different instance of the problem (eventually having the same employees as respondents in more than one instance). The goal is to predict, for each checkpoint, if a user is not favorable (score 1, 2, or 3) to a question that has not been answered yet. We know that there are several complex CF algorithms in the literature for implicit and explicit problems [Su and Khoshgoftaar 2009, Bokde et al. 2015]. However, to solve the addressed problem, we chose the simple and diverse set of CF algorithms described in Section 4, plus a linear ensemble trained over the outputs of the algorithms. In the present paper, instead of discussing which algorithm is the best one, we argue that it is possible to explore user-question interactions aiming to recognize answering patterns. Ultimately, these patterns can be used to build custom recommender systems.

For each given checkpoint, we perform a holdout technique: 20% of randomly chosen answers of each user are placed in a holdout set. The remaining data (80%) are used for training (70%) and validation (10%). The training data of 70% are then used for hyperparameter optimization performed by a 5-fold cross-validation approach with the validation data of 10% being used for early stopping training process. Each cross-validation procedure is performed independently over a specific checkpoint, possibly returning different hyperparameters. However, quite similar hyperparameters have been found in the experiments. Hence, the same hyperparameters of a given model (except for the matrix factorization model) have been used for all checkpoints. The matrix factorization model was found to perform better with larger latent dimensionality k for larger ts: $k = 25$, $k = 40$, $k = 45$ and $k = 50$ led to the best results for each checkpoint. All MF models were trained with L2-regularization $\lambda = 5$ with 10 to 15 epochs. For the NCF approach, it was found that $k = 64$, three fully connected hidden layers with sizes 64, 32 and 16, a learning rate $\alpha = 2 \times 10^{-4}$ and ReLU (Rectified Linear Unit) activation function (except to the output Logistic unit) returned best results. Only five epochs were required for convergence. For the logistic MF model, the best hyperparameters were: $k = 25$, $\lambda = 35$ and $\alpha = 2$. The item-item model has no hyperparameters, a notable advantage of this model.

A linear ensemble was also trained over the predictions of the individual CF models. Ensemble models are useful for handling imbalance effects, especially if applied over a set of heterogeneous models [He and Garcia 2009]. The input of this model is the predictions of the individual base models, while the output is the true scores. The linear model was trained using SGD with a batch size of 128, $\alpha = 0.01$, $\lambda = 0.5$ and MSE (mean squared error) as loss function. [Jahrer et al. 2010] shows more complex ensemble techniques applied to collaborative filtering algorithms.

The final models with the best hyperparameters were tested with the holdout set (20% of the data). The area under the ROC curve (AUC) evaluated for all algorithms and checkpoints are shown in Table 2. It measures the probability of a randomly chosen

negative example (scores 4 or 5) has a smaller estimated probability of belonging to the positive class (scores 1, 2, or 3) than a random positive example [Huang and Ling 2005]. Therefore, it is better than accuracy or other classification metrics since it is unclear how to define the ideal threshold between the negative and positive classes for this recommendation task. Moreover, it is better than MSE or other related metrics since they have poor performance for imbalanced problems.

Table 2. AUC values of five CF-based algorithms evaluated in the holdout set.

checkpoint	Item-item CF	ALS-MF	NCF	Logistic MF	Linear ens.
cp_1	0.833	0.858	0.815	0.818	0.845
cp_2	0.841	0.856	0.830	0.834	0.847
cp_3	0.859	0.872	0.837	0.846	0.864
cp_4	0.874	0.893	0.851	0.864	0.883

Results show that it is possible to *learn* the discrimination between low and high scores given by employees in large surveys, even in a very imbalanced setup. Even though no significant differences between performances have been found, a straightforward discussion can be made. The ALS-MF algorithm is the best-performing algorithm in all checkpoints, while the neural-based is the worst (in accordance with results of [Rendle et al. 2020]), performing very similarly to item-item. We point out that the results of the item-item algorithm are very satisfactory, given its conceptual simplicity and its fast and straightforward training procedure (compared with the others). Also, we believe that the great performance of ALS-MF is due to the use of bias terms that decouple the true behavioral signal (the one to be learned) from the noise patterns of the available ratings. Finally, it should be noted that NCF and logistic MF have been designed for implicit problems and thus suffer from not being able to discriminate zero ("0") values (high scores) from missing values. Another option would be excluding zero values from the training sets. However, the system would not be able to properly consider low scores due to poor evaluation (instead of missing values). Thus, we believe that the choice of merging them into one label introduces not relevant bias to the results. Otherwise, it would be possible to use hierarchical classification approaches for discriminating between all three cases (positive, negative and missing) which are out of scope.

The linear ensemble could not outperform ALS-MF, possibly due to the high correlation between the models' predictions, a remark observed from a correlation analysis. We notice that item-item and NCF have the most correlated outputs, with a Pearson coefficient of 0.96. The least correlated algorithm to others is the Logistic MF. If more diverse models were included into this pool, such as the models proposed by [Wu et al. 2019] and [Salakhutdinov et al. 2007], the ensemble result would be probably better.

6. Conclusions

This paper has addressed the problem of predicting missing answers on large surveys by using CF-based algorithms as predictor models. Five algorithms (item-item, iterative matrix factorization, neural collaborative filtering, logistic matrix factorization, ensemble with all of them) have been tested on a quite large survey with four checkpoints to provide the estimated matrix with pairs of respondent-answer events. In the experiments, the performances of the algorithms in terms AUC have been compared. Results showed that

there is no significant difference among algorithms performance and that there are patterns present in the scores given by employees. It should be noted that the results cannot reveal undesired effects such *satisficing* as they consider that favorable scores reflect the faithful perception of respondents. Nevertheless, this effect is partially mitigated by the bias terms of ALS-MF or the deviation modeling of item-item. An alternative approach could be using of comments of respondents (side information) for identifying this phenomenon. In future works, we intend to explore the learned patterns to reduce the questionnaire size and include other variables to be estimated like the commitment of each respondent to include or not comments about a specific aspect of the company.

Acknowledgments

M. Delgado acknowledges CNPq (grants 439226/2018-0, 314699/2020-1) for partial financial support.

References

Al-Shamri, M. Y. H. (2016). User profiling approaches for demographic recommender systems. *Knowledge-Based Systems*, 100:175–187.

Boim, R., Greenshpan, O., Milo, T., Novgorodov, S., Polyzotis, N., and Tan, W.-C. (2012). Asking the right questions in crowd data sourcing. In *2012 IEEE 28th International Conference on Data Engineering*, pages 1261–1264. IEEE.

Bokde, D., Girase, S., and Mukhopadhyay, D. (2015). Matrix factorization model in collaborative filtering algorithms: A survey. *Procedia Computer Science*, 49:136–146.

Chun, A. Y., Heeringa, S., and Schouten, J. (2018). Responsive and adaptive design for survey optimization. *Journal of Official Statistics*, 34(3):581–597.

Early, K., Mankoff, J., and Fienberg, S. E. (2017). Dynamic question ordering in online surveys. *Journal of Official Statistics*, 33.

Gonzalez, J. M. and Eltinge, J. L. (2008). Adaptive matrix sampling for the consumer expenditure quarterly interview survey. In *Proceedings of the Section on Survey Research Methods, American Statistical Association*, pages 2081–8.

He, H. and Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., and Chua, T. S. (2017). Neural collaborative filtering. pages 173–182. International World Wide Web Conf. Steering Committee.

Huang, J. and Ling, C. X. (2005). Using auc and accuracy in evaluating learning algorithms. *IEEE Transactions on knowledge and Data Engineering*, 17(3):299–310.

Jahrer, M., Töscher, A., and Legenstein, R. (2010). Combining predictions for accurate recommender systems. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 693–702.

Johnson, C. C. (2014). Logistic matrix factorization for implicit feedback data. *Advances in Neural Information Processing Systems*, 27(78):1–9.

Koren, Y. (2009). Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 447–456.

Krosnick, J. A. (2018). Questionnaire design. In *The Palgrave handbook of survey research*, pages 439–455. Springer.

Kulkarni, P. V., Rai, S., and Kale, R. (2020). Recommender system in elearning: a survey. In *Proceeding of International Conference on Computational Science and Applications*, pages 119–126. Springer.

Lavrakas, P. J. (2008). *Encyclopedia of survey research methods*. Sage publications.

Li, S., Karatzoglou, A., and Gentile, C. (2016). Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 539–548.

Lika, B., Kolomvatsos, K., and Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073.

Melville, P., Mooney, R. J., Nagarajan, R., et al. (2002). Content-boosted collaborative filtering for improved recommendations. *Aaai/iaai*, 23:187–192.

Ortigosa, A., Paredes, P., and Rodriguez, P. (2010). Ah-questionnaire: An adaptive hierarchical questionnaire for learning styles. *Computers & Education*, 54(4):999–1005.

Pazzani, M. J. and Billsus, D. (2007). Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer.

Rendle, S., Krichene, W., Zhang, L., and Anderson, J. (2020). Neural collaborative filtering vs. matrix factorization revisited.

Salakhutdinov, R., Mnih, A., and Hinton, G. (2007). Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798.

Schouten, B., Calinescu, M., and Luiten, A. (2013). Optimizing quality of response through adaptive survey designs. *Survey Methodology*, 39(1):29–58.

Song, L., Tekin, C., and Van Der Schaar, M. (2014). Online learning in large-scale contextual recommender systems. *IEEE Transactions on Services Computing*, 9(3):433–445.

Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009.

Wagner, J. R. (2008). *Adaptive survey design to reduce nonresponse bias*. PhD thesis, University of Michigan.

Wang, H., Wu, Q., and Wang, H. (2017). Factorization bandits for interactive recommendation. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Wu, C., Wu, F., An, M., Huang, J., Huang, Y., and Xie, X. (2019). Npa: Neural news recommendation with personalized attention. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2576–2584.

Wu, Q., Wang, H., Gu, Q., and Wang, H. (2016). Contextual bandits in a collaborative environment. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 529–538.

Zhang, C., Taylor, S. J., Cobb, C., and Sekhon, J. (2020). Active matrix factorization for surveys. *Annals of Applied Statistics*, 14.