A Comparative Evaluation of Top-N Recommendation Algorithms: Case Study with Total Customers

Sihem Amer-Yahia, Idir Benouaret

A Comparative Evaluation of Top-N Recommendation Algorithms: Case Study with Total Customers

1st Idir Benouaret
CNRS, Univ. Grenoble Alpes
Grenoble, France
idir.benouaret@univ-grenoble-alpes.fr

2nd Sihem Amer-Yahia
CNRS, Univ. Grenoble Alpes
Grenoble, Francey
sihem.amer-yahia@univ-grenoble-alpes.fr

***Abstract*—Industrial applications of recommendation systems aim at recommending top-N products that are the most appealing to their customers, often focusing on those products that customers are likely to purchase in the near future. In this experiments and analyses paper, we present an extensive experimental evaluation of various top-N collaborative filtering recommendation algorithms based on a real-world dataset of customer's purchase history provided by our business partners at TOTAL. Our study aims to compare representative collaborative filtering approaches in practice and study the ones yielding the highest recommendation accuracy, with respect to well-established evaluation measures. These experiments are part of the development of a promotional offers campaign for TOTAL customers owning a loyalty card. We show how different settings for training and applying the selected algorithms influence their absolute and relative performances. The results are valuable to our TOTAL partners as they constitute the first large-scale analysis of recommendation algorithms in the context of their datasets. In particular, the study of the impact of recency in the training set and the role of customer activity and of context in recommendation shed light on a finer design of promotional product campaigns.**

***Index Terms*—recommendation systems, evaluation**

I. Introduction

Recommendation systems are designed to help users automatically find relevant and desirable items in a large collection of items. In general, those systems work by means of predicting items that are likely to be the most appealing to users based on their preferences. Recommendation systems are widely deployed online in a variety of domains such as Google news personalization [1], Youtube videos [2], and Amazon products [3]. They also received a lot of attention in academia [4] with a particular interest in collaborative filtering (CF) [5], [6]. Unlike content-based recommender systems that require rich information about items to work well in practice [7], CF approaches have the advantage to solely rely on existing interactions between users and items. An interaction consists of a user's feedback on a specific item in the form of a rating, a purchase, a view, etc. Most research on CF exploits explicit feedback such as 5-star ratings, for example in the context of the Netflix[1] 1$ million prize. However, explicit ratings are not always available. In particular, in retail, customers do not usually provide an explicit rating on the products they purchase. In that case, the aim of a recommender system is to predict purchases of customers rather than their rating scores. When explicit ratings are not available, transactional datasets become binary and are referred to as *implicit feedback* datasets. An important challenge about implicit feedback such as purchases and clicks is that only positive user-item interactions are available. The missing data is ambiguous and is a mixture of real negative feedback and unknown values [8], [9]. For example, if a customer has never purchased a product, she might not like it, not be aware of it, or the product might be out of stock. This direction has not received much attention in contrast to rating prediction. That is likely due to the lack of publicly available datasets as it is very sensitive to retailers to release their purchase datasets.

In this paper, we present an experimental case-study on various recommendation algorithms using data provided by our partners at TOTAL. The dataset contains the purchase history of more than $440,000$ customers owning a loyalty card, over a period of almost 3 years, and containing about 3 million purchases. We systematically implement and evaluate four families of CF approaches: association rule mining [10], [11], item-based CF [12], matrix-factorization [8], and Bayesian Personalized Ranking [13], and make the following contributions:

1) We show that all personalized algorithms perform better than a non-personalized baseline that was being used so far by the marketing department at TOTAL.
2) When working with the entire dataset, we find that item-based CF is superior to all other recommendation approaches. It is relevant to point out that our results are not entirely consistent with those reported in other case studies, such as [14] which found that the simplest bi-gram association rule model performs better than other more established collaborative filtering approaches including neighborhood and matrix factorization models. This implies that no general conclusion can be drawn about the relative performance of each recommendation

[1] https://www.netflixprize.com/

algorithm without conducting extensive experiments on the real-world datasets.

3) To examine the effect of recency of purchases on accuracy, we run experiments with training sets of different lengths: last 18 months, last 12 months and last 6 months. Our results clearly show that training on all available data is inferior to training only on recent purchases, which tells us about the importance of incorporating recency in generating recommendations in retail. Our findings suggest that the most recent training history (6 months) yields the highest accuracy. Similar observations were reported in the state of the art on other datasets and application domains [15], [16], [17].
4) We continue our exploration of the effect of time on recommendation accuracy and run experiments that split the training set into morning, afternoon and evening. We observe that while item-based CF remains superior, significant prediction accuracy improvements can be achieved by incorporating temporal contextual information into recommendation algorithms. Similar improvements can be achieved with location context, where were build a `local` model for each of the projected purchases in a given region and compare it to a `global` model which is trained on the whole training set.
5) The long tail nature of customer purchases led us to verifying the performance of the algorithms on different customer segments according to their purchase frequency. Our results show a clear difference in performance, and highlight several types of behaviors where each one is better addressed by a particular CF recommendation algorithm. In particular, we show that for rare customers who constitute the bulk of our customer base, relying on association rule mining yields the highest accuracy as it finds correlations with those customers' future purchases. Frequent customers however are best served by matrix-factorization that is known to work well with a rich history. For occasional customers, item-based CF is shown to perform best. This indicates that no general conclusion can be drawn on the relative performance of each algorithm, and that testing all methods with different customer segments is mandatory. This suggests that in a real industrial setting like ours, it is necessary to adapt and select the right recommendation algorithm according to the segment that each customer belongs to.
6) Finally, our marketing partners suggested to examine recommendation accuracy when treating all products in one category as the same product (e.g., *Lays chips* and *Kettle chips*). We find that recommendation accuracy improves for all algorithms while maintaining the initial result where the item-based collaborative filtering is best. This is an optimistic case that suggests a finer categorization of products in the future and a study of its effect on recommendation accuracy.

The results of this work are highly valuable to the marketing department in TOTAL and are currently guiding product managers design promotional offers in different contexts.

The rest of paper is organized as follows. In Section II, we present our dataset and its characteristics, as well as the recommendation task behind this study. In Section III, we describe our experimental protocol and evaluation metrics. We then give a brief theoretical foundation of the recommendation algorithms we used in this study. In Section V, we report and analyze our experimental results in various settings. Related work is surveyed in Section VI. Finally, Section VII concludes the paper.

II. Dataset and Recommendation at TOTAL

TABLE I
Summary of the characteristics of our filtered dataset

	TOTAL dataset
Domain	retail, gas and oil industry
Time span	Jan 2017 –>Sept 2019
Number of customers	$442,520$
Number of products	$9,366$
Number of interactions	$2,833,938$
Sparsity	99.93%

This experimental study is part of our research project with the marketing department of TOTAL[2]. The company manages $3,472$ stations in France, and sells mostly *gas* and other products such as car services (*car wash, oil change, lubricant, etc*), *drinks* and *food* products. TOTAL expressed a need for an automatic recommendation system to help their customers find interesting products that they do not necessarily know, in order to increase customer satisfaction and keep them away from competitor retailers. The deployment scenario chosen by our business partners is to first design and evaluate a set of recommendation algorithms in different contexts, and then to choose the right algorithm and setting for an actual deployment in gas stations and for running promotional offers as well. This paper focuses on the first phase where various recommendation algorithms are evaluated in different settings, and compared to the non-personalized baseline (`MostPop`) they have been using, consisting in recommending the most popular products that customers have not already purchased.

Our dataset represents customers purchasing products at different gas stations that are geographically distributed in France, for a period of 2 years and 9 months (from January 2017 to September 2019). The dataset $\mathcal{D}$ is represented as a set of records of the form $\langle id, c, p, t\rangle$, where id is a unique receipt identifier, c is a customer, p is a product purchased by c and t corresponds to the timestamp of the transaction. The set of products were filtered out by business constraints. We removed products marked as "non-interesting ", according to the marketing department, such as *gas, plastic bags, clothes, high tech products*. We also removed products that are related to promotional offers which were either given for free or purchased at a discounted price as those products do not

[2] TOTAL S.A. is a French multinational integrated oil and gas company founded in 1924 and one of the seven ”supermajor” oil companies in the world

TABLE II
PURCHASE HISTORY OF AN ANONYMIZED CUSTOMER AND TOP-5 PRODUCT RECOMMENDATIONS ACCORDING TO SELECTED COLLABORATIVE FILTERING ALGORITHMS. PRODUCT DESCRIPTIONS WERE TRANSLATED TO ENGLISH.

	Top-5 recommendations for each algorithm for the same customer			
Customer's purchase history	ARM	IBCF	Implicit-ALS	BPRMF
TOTAL deicer *Orangina 33cl* *Winter windscreen washer* *Ham & cheese Pizza* *Manhattan salad* *Engine oil 4tz* *Brake fluid hbf4* *Coca Cola 1.5L*	*Evian sparkling 1L* *TOTAL windshield washer* *Cristaline water* *TOTAL car wash* *Expresso*	*TOTAL windshield washer* *TOTAL car wash* *Cristaline sparkling* *Coca Cola 50cl* *Lg bug remover*	*TOTAL windshield washer* *Coca Cola 50cl* *Salad Roma 320* *Salad Antibes* *Evian water 75 CL*	*TOTAL car wash* *TOTAL Adblue* *TOTAL windshield washer* *Plastic wipes* *Lg bug remover*

always reflect customer preferences. This led to a dataset whose basic characteristic are summarized in Table I. It consists of $442,520$ loyal customers and $9,366$ products, with $2,833,938$ corresponding customer × product interactions. This yields a very high level of the sparsity of the customer × product interaction matrix: about 99.93% of all interactions are missing which makes the recommendation task more difficult as a high sparsity is a critical issue in that context [18].

Table II shows the purchase history of a randomly selected customer, as well as a list of top-5 recommendations that were computed with different CF algorithms that we present in Section IV. One can see that each algorithm produces a different recommendation list.

III. EXPERIMENTAL PROTOCOL

A. Purchase Matrix

Let $\mathcal{U} = \{u_1, u_2, ..., u_m\}$ be the set of all customers and $\mathcal{I} = \{i_1, i_2, ..., i_n\}$ be the set of all products. For a given customer u, $H_u \subseteq \mathcal{I}$ denotes the purchase history of u , the set of all products ever purchased by u. The training stage of the algorithms we evaluate takes as input a *purchase matrix*, where each column corresponds to a product and the customers that have purchased it, and each row represents a customer and the products she purchased. We denote $\boldsymbol{P}$ the *purchase matrix* of the m customers in $\mathcal{U}$ over the n products in $\mathcal{I}$. An entry $p_{u,i}$ in the matrix contains a boolean value (0 or 1), where $p_{u,i} = 1$ means that product i was bought by customer u at least once (0 means the opposite).

B. Evaluation Protocol

The widely used strategy for evaluating recommendation accuracy in offline experiments is to randomly split the data into training and test sets. However, this setting does not reflect well the reality in the retail context as it is time agnostic. The availability of timestamps in the purchase records enables us to attempt a more realistic and valuable experiment. We hence train our algorithm on past purchases and test the recommendations on future purchases. We split the dataset according to a given point in time which acts as our "present" (the time we apply our algorithm). Our marketing partners suggested to use purchase records from January 2017 to December 2018 for training the algorithms and records from January 2019 to September 2019 for testing them.

C. Evaluation Metrics

For all test customers each algorithm outputs a sorted list of top-N products. Recommendation lists are then evaluated for each test customers using both an accuracy measure: $F1$-score and a ranking measure: Discounted Cumulative Gain (DCG). For a given customer u we note T_u the set of items that are purchased in the test set. A recommended product i is relevant if i to the target set T_u. $F1_u@N$ and $DCG_u@N$ are defined as follows:

$$F1_u@N = \frac{2.Precision_u@N.Recall_u@N}{Precision_u@N + Recall_u@N} \tag{1}$$

where precision and recall are defined as:

$$Precision_u@N = \frac{|R_u@N \cap T_u|}{N}$$

$$Recall_u@N = \frac{|R_u@N \cap T_u|}{|T_u|}$$

where T_u is the target set and $R_u@N$ is the of top-N list.

$$DCG_u@N = \sum_{i=1}^{N} \frac{rel_i}{log_2(i+1)} \tag{2}$$

where $rel_i = 1$ if $i \in T_u$ and 0 otherwise.

To compute the final performance values, we average all metrics over all test customers. The value of the cutoff N is chosen by our business partners and is set to $N = 10$. They specified that in a real application scenario, 10 recommendations will be displayed for each customer. Results are converted to the range [0%, 100%] (see Section V) for a better readability.

IV. RECOMMENDATION ALGORITHMS

A. Bi-gram Association Rules

Association rules mining [19] is one of the most frequently used techniques to analyze customers' purchasing patterns. Recommendation systems can benefit from association rules [10], [11]. As shown in the experimental study by Pradel *et al.* [14], association rules have demonstrated good performance in recommendations using real-world e-commerce datasets. We leverage the purchase history of our customers to extract association rules of the form $i \Rightarrow j$. We use these *bigram* rules to compute an association matrix

$\boldsymbol{A}$ between each pair of products i, j, where each entry $a_{j,l}$ corresponds to the confidence of the association rule $j \Rightarrow l$.

$$conf(j \Rightarrow l) = \frac{P_{\bullet j}^T P_{\bullet l}}{||P_{\bullet j}||_1}, ||P_{\bullet j}||_1 = \sum_{i=1}^{n} |p_i| \tag{3}$$

Where $P_{\bullet i}$ is the i-th column of the purchase matrix $\boldsymbol{P}$, and $\boldsymbol{X}^T$ is the transpose of matrix $\boldsymbol{X}$.

Therefore, to generate top-N recommendations for customer u, we first identify a set of rules that are supported by the purchase history of u. i.e., rules of the form $k \Rightarrow l$, where k is purchased by u. Then, non purchased products are ranked either by their maximum confidence [10], [14], or the sum of confidences [11] of all rules. In our case, the sum aggregation was found to give slightly better results because it takes into account the whole purchase history of the customer to compute the prediction for a given product. Thus, we compute the score of a product j for a customer u as follows:

$$score(u, j) = \sum_{i \in H_u} conf(i \Rightarrow j) \tag{4}$$

Where, H_u is the purchase history of customer u, and j is a candidate product for recommendation. Products are then sorted according to their respective scores and the top-N products are recommended to u.

B. Item-based Collaborative Filtering

This algorithm takes as input the training purchase data and computes a similarity matrix between products. Following [3], we compute similarity between a pair of products i and j using the cosine between their corresponding column vectors in the purchase matrix. More formally,:

$$sim(i, j) = \frac{P_{\bullet i}^T P_{\bullet j}}{\sqrt{P_{\bullet i}^T P_{\bullet i}}\sqrt{P_{\bullet j}^T P_{\bullet j}}} \tag{5}$$

The output of the training phase is the similarity matrix between products $\boldsymbol{S}$, where the $(i, j)^{th}$ entry in the similarity matrix $\boldsymbol{S}$ s_{ij} is equal to $sim(i, j)$.

The top-N recommendations for a customer u is computed as follows. For each product k in H_u, we retrieve its similarity vector s_k using the similarity matrix $\boldsymbol{S}$. We obtain $|H_u|$ similarity vectors of products with similarity scores for each product in the purchase history of customer u. We then average those vectors s_k to get a final similarity vector s and we select from s the N products that have the highest similarity scores.

C. Matrix Factorization

Matrix factorization maps users and items to a joint latent factor space of dimensionality r, such that user-item interactions are modeled as inner products in that latent space [20], [21]. More formally, each user is associated with a vector $x_u \in \mathbf{R}^r$ and each item i is associated with a vector $y_i \in \mathbf{R}^r$. The resulting inner product, $x_u^T y_i$ captures the estimated preference of user u for item i. Koren *et al* extended their approach for the case of implicit feedback datasets [8]. In this case, the matrix consists of estimates of preferences induced by customer interactions such as clicks, views, etc instead of ratings. In our case, we use the number of times a customer u purchased a product i as the implicit rating, denoted r_{ui}. A set of binary variables p_{ui} indicating the preference of product i with respect to customer u are introduced:

$$p_{ui} = \begin{cases} 1 & r_{ui} > 0 \\ 0 & r_{ui} = 0 \end{cases} \tag{6}$$

These binary preferences are associated with varying confidence levels. The number of times a customer purchased a product depicts his or her interest over the product. To model this behavior, confidence values are constructed using a predefined constant α that captures a purchase rate. The idea is that as r_{ui} grows, we have a stronger indication that the customer effectively likes the product. Thus, the confidence c_{ui} in estimating p_{ui} is defined as follows:

$$c_{ui} = 1 + \alpha \times r_{ui} \tag{7}$$

The confidence is calculated using the magnitude of the implicit ratings r_{ui}, giving us a larger confidence when a product is purchased many times by the same customer. The rate of increase in confidence is controlled by a constant α, which is data-dependent and thus determined by a grid search over a set of values.

The goal now, is to find a vector $x_u \in \mathbf{R}^r$ for each customer u, and a vector $y_i \in \mathbf{R}^r$ for each product i that will factor customer preferences.

$$min \sum_{u,i} c_{ui}(p_{ui} - x_u^T y_i)^2 + \lambda \left(\sum_u ||x_u||^2 + \sum_i ||y_i||^2 \right) \tag{8}$$

The term $\lambda \left(\sum_u ||x_u||^2 + \sum_i ||y_i||^2 \right)$ in Equation 8 is a necessary regularization parameter to avoid over-fitting the training data. The Alternating Least Squares method is used for the optimization [22] of the loss function and once the user and item vectors are computed, preferences are estimated as inner products: $\hat{p}_{ui} = x_u^T y_i$.

To generate the top-N recommendations for a customer u, all products i are sorted by decreasing scores of $\hat{p}_{ui}$ and the top-N products are recommended to u.

D. Bayesian Personalized Ranking

Bayesian Personalized Ranking [13] is an optimization principle for CF methods, designed explicitly to deal with implicit feedback datasets. This method falls into the category of " learning-to-rank" methods as a general framework for pairwise learning. Different from matrix-factorization methods, it uses item pairs as training data and optimizes for correctly ranking item pairs instead of estimating scores for single items. This assumes that if a user u has expressed an implicit preference such as a purchase on an item i, then u prefers this item over all other non-observed items. Hence a training instance in our case is a triple (u, i, j), where we assume that customer u prefers product i over j, that we note $i >_u j$. The set of all inferred preferences $\mathcal{D}_S$, i.e., the training data used for optimization, is defined as follows:

$$\mathcal{D}_S = \{(u,i,j)|i \in H_u \land j \in \mathcal{I} \setminus H_u\} \tag{9}$$

The generic optimization criterion is given as:

$$OPT(\mathcal{D}_S) = argmax_\Theta \sum_{(u,i,j)\in\mathcal{D}_S} ln\, \sigma(\hat{x}_{u,i,j}) - \lambda_\Theta ||\Theta||^2 \tag{10}$$

Where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the logistic sigmoid function, $\hat{x}_{u,i,j}$ is the pairwise prediction for user u and items i, j, Θ is a parameter vector of an arbitrary model and λ_Θ is a model specific regularization parameter to prevent over-fitting the model.

$\hat{x}_{u,i,j}$ is a real-valued function of Θ which captures the relationship between customer u, product i and product j. The estimation of $\hat{x}_{u,i,j}$ is performed through matrix-factorization but since it can only predict single scores, the estimator is decomposed into single prediction tasks: $\hat{x}_{u,i,j} = \hat{x}_{u,i} - \hat{x}_{u,j}$. The optimization is performed using Stochastic Gradient Descent with bootstrap sampling of training triples using the following update rule [13]:

$$\Theta \leftarrow \Theta + \alpha \left(\frac{e^{-\hat{x}_{u,i,j}}}{1+e^{-\hat{x}_{u,i,j}}} . \frac{\partial}{\partial\Theta}\hat{x}_{u,i,j} + \lambda_\Theta . \Theta \right) \tag{11}$$

where α is the learning rate.

V. Experiments

All our implementations are in Python 3.7.0 running on a 2.7 GHz Intel Core i7 machine with a 16 GB main memory and OS X 10.13.6. We first report a standard evaluation where we compare the overall performance of all algorithms (Section V-C). We then examine how the recency of the training set affects accuracy (Section V-D). We dive deeper into the temporal and spatial question in Section V-E and study how accounting for different temporal contexts affects performance. We then report a brief experiment that accounts for product categories in computing recommendation accuracy (Section V-G).

A. Implemented Algorithms

The algorithms we implemented with their corresponding model parameters are summarized. Association rules mining CF (Section IV-A) and Item-based CF (Section IV-B) are implemented from scratch. For the implicit matrix factorization approach (Section IV-C), we use the implementation provided in the MLlib package [23] of Apache Spark[3]. For the Bayesian Personalized Ranking approach (Section IV-D), we rely on the implementation provided in the *Implicit* library[4] by Ben Frederickson. We also integrate a most popular recommendation as a baseline. We use the following acronyms to refer to the algorithms:

- ARM: the bi-gram association rule mining recommender, described in Section IV-A;
- IBCF: the item-based collaborative filtering approach, described in Section IV-B;
- Implicit-ALS: described in [8] and presented in Section IV-C. We set the number of latent factors $r = 40$, the confidence constant is set to $\alpha = 10$ and the regularization parameter is set to $\lambda = 0.001$;
- BPRMF: described in [13] and presented in Section IV-D. We set the number of latent factors $r = 20$, the learning rate $\alpha = 0.001$ and the regularization parameter $\lambda = 0.005$;‘
  We note that for both Implicit-ALS and BPRMF we report the best parameter values that we obtained using a grid search.
- MostPop: to mimic the approach used at TOTAL, we implemented a popularity-based algorithm that recommends the most popular products excluding previously purchased ones. Despite its simplicity, it is often a strong baseline in retail.

B. Dataset Preparation

As it often practiced in the recommendation literature [8], [13], for each experiment, we discard customers who purchased fewer than 5 products in the training set. An important characteristic of our dataset and of all datasets from the retail domain is the tendency to repetitively purchase the same products at different times. It is however much more valuable for the customer and even for the retailer to recommend products that the customer has not purchased recently, or is not aware of. In addition, we noticed that if we simply randomly select N products from the purchase history of each customer as the top-N recommendations, we can reach reasonable results. Thus, after several exchanges with the marketing department at TOTAL, for each test customer we decided to remove the “easy” predictions from the test set corresponding to the products that have been purchased by that customer during the training period. This setting makes the task of predicting the correct products harder but potentially more impactful in a real-world scenario.

C. Standard Evaluation

This setting corresponds to the case where each row of our training purchase matrix contains all known purchases of training customers before the split date at which training and test sets are separated: January 1^{st}, 2019. All algorithms are evaluated using exactly the same test customers and the corresponding target sets.

The values of $F1@10$ and $DCG@10$ for various algorithms are shown in Table III. We can see that all personalized algorithms perform better than the non-personalized baseline MostPop. It can also be noted that IBCF achieves the best results on all metrics, and performs just slightly better than Implicit-ALS, as the differences in their relative performances are not very significant. We also find that both

[3] https://spark.apache.org/docs/latest/mllib-collaborative-filtering.html

[4] https://implicit.readthedocs.io/en/latest/bpr.html

TABLE III
TEST RESULTS FOR THE COMPLETE HISTORY SETTING. ALGORITHMS ARE TRAINED ON THE WHOLE CUSTOMER × PRODUCT PURCHASE MATRIX. BEST ACCURACY IS REPRESENTED IN BOLD

Recommender	$F1$@10	DCG@10
`ARM`	6.76%	55.77%
`IBCF`	**8.06%**	**63.71%**
`Implicit-ALS`	7.81%	60.05%
`BPRMF`	5.96%	42.91%
`MostPop`	6.02%	39.32%

`ARM` and `BPRMF` achieve a low performance compared to `IBCF` and `Implicit-ALS`.

It is relevant to point out that our results are not entirely consistent with those reported in other case studies, such as [14] which found that the simplest bi-gram association rule model performs better than other more established collaborative filtering approaches including neighborhood and matrix factorization models. This implies that no general conclusion can be drawn about the relative performance of each recommendation algorithm without conducting extensive experiments on real-world datasets.

The rather low performance of `BPRMF` is surprising, especially that this approach is specifically designed for implicit feedback datasets. A possible explanation lies in the choice of the model parameters. Following Rendle et al. [13], we used grid search to choose the best parameter setting in every experiment. We can however not exclude that there exists some parameter combinations outside the ones we tested for which `BPRMF` may perform better. Some recent works [24] and [25] improve over `BPRMF` to account for the integration of heterogeneous feedbacks such as clicks and add-to-cart. However, in our dataset, the only available feedback is the purchase or not of a product which renders the latest findings inapplicable.

D. Contribution of Recency

Since customers' preferences may drift over time, we carried out a set of experiments with various sizes of the training history and measured the effect of recency of the training set on accuracy. To this end, we split the training data chronologically into different training sets using an expanding time window approach. This partitioning is performed on a 6-month basis. The test set still contains all customers' purchase records from January 2019 to September 2019. But now, we evaluate recommendation performance using three different training sets corresponding to three sizes of the training purchase history. We used a 6-month period, a 12-month period and an 18-month period of the available training data before the split date. We note that the algorithms are tested using exactly the same test customers and the same protocol as the complete history setting.

Performance results are shown in Table IV for the 6-month, 12-month and 18-month training period setting. The values between parentheses indicate the gain in performance over the default 24-month training period reported in Section V-C. Compared to the complete history setting, we can see that in general considering the most recent purchases as the training data has a positive impact on all performance measures and all recommendation algorithms. The smallest improvement is for `Implicit-ALS` since matrix factorization is less sensitive to variations in the training sets. The best results are obtained when training the algorithms with the purchase data that occurred within 6 months before the split date.

These findings suggest that the most recent training history yields the highest accuracy. Similar observations were reported in the state of the art on other datasets and application domains [15] , [16], and [14]. In those works, it was found that recency matters in capturing evolving users' tastes. This is also an important factor in our context, for example a customer who changed her car will probably drift in her purchasing behavior, as some products that she used to purchase might not be suitable anymore for her new car.

Our results provide additional evidence that the recency of customer feedback plays an important role in CF algorithms. However, it appears to be more pronounced for `ARM` as the extracted rules are more accurate and have a higher confidence when training only on most recent purchases. A similar behavior is observed for `IBCF` which achieves the best performance values over all algorithms when considering only the 6 most recent months for training (see Table IV). Algorithms based on latent factors models (`Implicit-ALS` and `BPRMF`) are less sensitive to the recency of customer feedback and are more consistent when training on different sizes of the training sets. For instance, the relative differences between the performances of `Implicit-ALS` are not very significant with respect to the 3 different settings. The above results are mainly due to how `ARM` and `IBCF` work, as they consider one product at a time, and generate a list of products that are highly associated or that are highly similar to that product. Thus, they are more sensitive to the purchase recency, whereas matrix factorization approaches (`Implicit-ALS` and `BPRMF`) dilute recency information when estimating customer factors.

Table V-C show an example of Top-10 recommendation for a customer using the different history settings. This shows that training on 6 months achieves the best performance with 5 hits in the top-10.

E. Contextual Recommendations

This experiment aims to verify the assumption that since customers purchase different products at different times and different places, contextual information influences recommendations. We will show that incorporating contextual factors into the recommendation process leads to better recommendation performance.

1) Temporal Context: Figure 1 shows the number of customer visits (i.e., purchasing products) for each hour of the day. A clear pattern in the purchase behavior is visible throughout the day. As expected, customer activity is limited during the night, for example, only $4,586$ customers visited one of the stations between 3AM and 4AM, which represents only about 1.03% of all customers. In the morning at 6AM, the amount of activity starts to increase as people usually tend to go to work

TABLE IV
TEST RESULTS FOR DIFFERENT HISTORY SETTINGS. BEST RESULTS ARE PRESENTED IN BOLD

	6-month history setting		12-month history setting		18-month history setting	
Algorithm	*F*1@10	DCG@10	*F*1@10	DCG@10	*F*1@10	DCG@10
ARM	9.04% (+33.77%)	60.54% (+8.55%)	8.01% (+18.49%)	57.89% (+3.80%)	7.64% (+13.01%)	57.06% (+2.31%)
IBCF	**10.47%**(+29.9%)	**72.94%**(+14.48%)	8.97% (+11.29)	68.51% (+7.53%)	8.75% (+8.56%)	64.39% (+1.06%)
Implicit-ALS	8.35% (+6.91%)	65.95%(+9.82%)	7.92% (+1.4%)	63.17% (+5.19%)	8.6%(+10.11%)	62.87% (+4.68%)
BPRMF	7.42% (+23.48%)	45.11% (+5.12%)	6.14% (+3.02%)	44.86% (+4.54%)	6.47% (+8.55%)	44.16% (+ 2.91%)
MostPop	7.62% (+26.57)	43.62% (+10.93%)	6.74% (+11.96)	41.61% (+5.82%)	6.46% (+11.96)	40.43% (+2.82%)

TABLE V
TOP-10 RECOMMENDATIONS FOR THE SAME CUSTOMER USING IBCF WITH DIFFERENT TRAINING SETTINGS – HITS AGAINST THE TEST SET (LAST COLUMN) ARE HIGHLIGHTED IN BOLD IN EACH SETTING.

IBCF (6-month)		IBCF (12 months)		IBCF (18 months)		Test set
history	Top-10 (**5 hits**)	history	Top-10 (**3 hits**)	history	Top-10 (**2 hits**)	
TW Lave glace hiver 4L *Total Quartz 5W30 1L* *Total Wash 15*	*TW Lave glace* **Lg Demoustiqueur** *TW Lave glace -20 4L* **Total Wash 25** *Total Adblue 1* **Total Wash 65** *Lg Degivrant -30C 4L Op* **Total Adblue 5L** *16 Lingettes Plastic* **Total Wash 35**	*Haribo World Mix* *Koala Lait Lutti* *Tarte Frambo* *TW Lave Glace* *TW Lave Glace Hiver 4L* *Sdw Bag Mega Thon Oeuf* *SDW mlx plt cru* *total Quartz 5W30 1L* *Mars X 3 135G* *Bounty Lait X 6 171G* *coca cola slim 33CL*	**Lg Demoustiqueur** **TW,lave glace -20 4L** *Evian 1L PET* *Cristaline Pet 1,5L* *Coca Cola Pet 50Cl* *TW Lave glace* *SDW mega tom.270G* *Total Adblue 1* ***Total Wash 25*** *Evian 75 CL*	*Plateau Repas* *kit kat X* *Haribo World Mix* *Koala Lait Lutti* *Tarte FramboO* *16 Lingettes Plastic* *TW lave glace* *TW Lave glace hiver 4L* *Total Adblue 5L* *Sdw Bag Mega Thon Oeuf* *ELF SELFMIX 2 S* *Navette Jambon Beurre* *SDW mlx plt cru* *Total Quartz 5W30 1L* *MARS X 3 135G* *Bounty Lait X 6 171G* *Lotus Gaufres* *coca cola slim 33CL* *ampoule H4 BL 2*	**Lg Demoustiqueur** *Evian 1L PET* *Total Adblue 1* **TW Lave glace -20 4L** *Cristaline Pet 1,5L* *TW Lave glace* *Coca Cola Pet 50Cl* *Lg Degivrant* *Evian 75 CL* *SDW jamb emme 130G*	*TOTAL Wash 35* *Total Adblue 5L* *TW lave glace -20 4L* *Adblue 10L* *Lg Demoustiqueur L* *Total LIQ.Ref 1L* *TotaL Wash 25* *TOTAL,LIQ.REF 5L* *TOTAL Wash 65* *Code lavage MP2* *Butane 6KG*

TABLE VI
RESULTS FOR TEMPORAL CONTEXT

	Context = Morning				Context = Afternoon				Context = Evening			
	Non contextual		Contextual		Non contextual		Contextual		Non contextual		Contextual	
Algorithm	F1@10	DCG@10	F1@10	DCG@10	F1@10	DCG@10	F1@10	DCG@10	F1@10	DCG@10	F1@10	DCG@10
ARM	6.19%	43.22%	6.74%	47.73%	4.88%	40.86%	6.47%	45.59%	4.63%	34.74%	6.24%	38.85%
IBCF	7.03%	46,97%	**8.34%**	**50.48%**	5.79%	52.76%	**8.65%**	**59.05%**	6.73%	55.07%	**7.46%**	**58.98%**
Implicit-ALS	6.25%	44.53%	7.77%	49.97	5.38%	47.66%	7.32%	52.68	6.68%	52.77%	7.19%	54.63%
BPRMF	5.05%	32.27	5.71%	41.04	4.41%	29.62%	5.23%	39.15%	4.72%	23.74%	4.56%	31.28%
MostPop	4.76%	24.59	5.10%	32.58	3.82%	22.75%	4.92%	31.03%	2.84%	18.61%	4.17%	24.39%

TABLE VII
CONTEXT = *Ile-de-France*

	ARM		IBCF		Implicit-ALS		BPRMF		MostPop	
Measure	Local	Global	Local	Global	Local	Global	Local	Global	Local	Global
F1@10	8,01%	5,89%	**9,4%**	7,11%	8,51%	7,3%	5,39%	5,25%	6,32%	5,04%
DCG@10	42.76%	35.45%	**61.17%**	52.86%	59.53%	51.85%	32.49%	25.86%	24.06%	19.13%

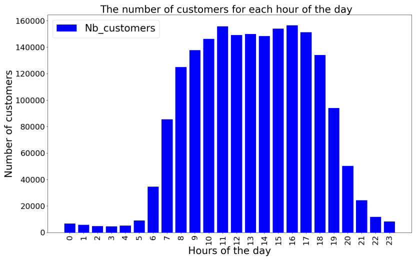

Fig. 1. The number of customer visits partitioned according to the hour of the day for the whole dataset (January 2017 – September 2019)

between 6AM and 9AM and pass by a gas station on their way. The number of different customers visiting each station reaches its peak at 11AM-12PM and remains high until 6PM, before decreasing to only 8,284 customers between 11PM and 11:59PM. In addition to the frequency of visits, the temporal context may also influence the customers' purchasing patterns. A customer might have different preferences according to the time of day the customer visits a station. As an example, a coffee drinker would be happy if recommended a coffee in the morning but maybe not as happy in the evening.

Following the context-aware multidimensional model in [26], we evaluate the performance of the recommendation algorithms as if we knew the context at predicting time, i.e., we recommend products simulating a real visit of a customer according to her time of visit. In the simplest form of the context-aware model, given the context in which a recommendation is performed, the prediction is based solely on customers' past purchases that happened within that same context. To this end, we created three different temporal contexts: *Morning*, *Afternoon* and *Evening* according to purchases that occurred between 6AM-12PM, 12PM-6PM, 6PM-12AM, respectively. We did not use purchases during the time span 12AM-6AM because of their very low number.

- During the training step, we build 3 models for each of the algorithms, one for each defined context. Each of these models is trained on a specific projection of the purchase matrix according to purchases that happened in a specific context.
- To validate the algorithms for different contexts, we split the test set into three distinct sets that happened at different times, and compute accuracy separately in each case.

Tables VI report the results for both the non-contextual and contextual settings. We observe that significant prediction accuracy improvements are achieved by incorporating contextual information into recommendation algorithms. `IBCF` remains superior to other approaches. Moreover, it achieves higher accuracy than training on the full history (Table III). This result makes a strong case for accounting for context in both training and test sets.

2) Location Context: Purchasing patterns are different in different locations. For instance, in our analysis, we found that *Ice cream* products are mostly consumed in the region around Paris and in the South of France, and that *Hot drinks* are less attractive in the south of France. Following the same approach that is described in Section V-E1, we evaluate the performance of the recommendation algorithms as if we know the context at predicting time, i.e., in this case, we recommend products simulating a real visit of a customer to a gas station according to the region where the transaction occurred. To this end, we created 13 different geographical contexts, where each context corresponds to one of the 13 French regions (e.g., *Auvergne-Rhônes-Alpes*), according to purchases that occurred in each region.

- During the training step, for each algorithm we build 13 different `local` models, one model for each location context. We also split the test set into 13 different disjoint test sets.
- To validate the algorithms for each context, we compare the achieved performance of each `local` model against a `global` model which consists of training on all available purchases.

Tables VII report the results for both the `local` and `global` model for each of the implemented algorithms for the French region: *Ile-de-France*. We observe that there is a significant improvement of recommendation performance when incorporating location context into the recommendation process for all algorithms, meaning that for predicting purchases in a specific region, we achieve a higher accuracy when training on the contextualized dataset (`local` model) compared to training on the full dataset (`global` model). The results for other regions are omitted due to lack of space, but similar results are obtained across all regions.

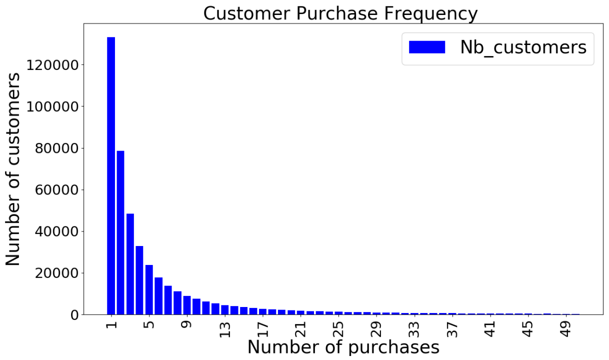

Fig. 2. Frequency of customer purchases

TABLE VIII
NUMBER OF CUSTOMERS FOR EACH SEGMENT AND THEIR CORRESPONDING PROPORTIONS

Customer Segment	Size	Proportion wrt to full dataset
Rare	257,051	70.04%
Occasional	88,815	24.20%
Frequent	21,069	5.75%

F. Customer Segmentation

This experiment examines the performance of our algorithms for different customer segments. A segment is defined by the number of purchases made by the customer before the split date. Our aim is to analyze the accuracy of each algorithm for different customer segments and to verify if the previously provided results in the standard evaluation hold or differ when considering the purchase frequency of our customers.

To better understand the distribution of purchases, we plot the purchase frequency of customers in Figure 2. The plot shows that about 30% of customers completed only one transaction. As usual in retail, our dataset exhibits a severe long tail effect [27]. This distribution raises the question that the frequency of purchases may play a role in recommendation accuracy and that the results we obtained in Section V-C might not hold for different customer segments.

Table VIII contains the customer segments we define and their corresponding sizes and proportions with respect to the full training set. The **Rare** segment contains customers with fewer than 4 purchases who correspond to those who acquired a loyalty card shortly before the split date: January 1^{st}, 2019. The **Occasional** segment includes customers who made between 5 and 20 purchases. The **Frequent** segment corresponds to customers with over 20 purchases.

Tables IX show our results for $F1@10$ respectively, for every customer segment. In summary, our findings are: `ARM`

TABLE IX
Test results of $F1$@10 for each customer segment

Segment	nb purchases	F1@10				
		ARM	IBCF	Implicit-ALS	BPRMF	MostPop
Rare	1	**7.03%**	6.19%	4.82%	5.23%	6.86%
	2	**7.96%**	7.15%	5.93%	5.36%	6.74%
	3	**8.07%**	7.16%	6.21%	6.11%	6.61%
	4	**8.6%**	8.08%	6.61%	6.76%	6.85%
Occasional	5-10	8.97%	**9.22%**	6.73%	5.9%	6.59%
	11-20	7.92%	**9.05%**	8.23%	6.12%	6.28%
Frequent	21-40	7.05%	7.97%	**8.72%**	7.13%	5.62%
	>40	5.69%	6.67%	**7.78%**	5.61%	4.77%

TABLE X
Test results of DCG@10 for each customer segment

Segment	nb purchases	DCG@10				
		ARM	IBCF	Implicit-ALS	BPRMF	MostPop
Rare	1	**46,87%**	38.81%	31.58%	42.67%	45.13%
	2	**54.89%**	54.47%	32.84%	30.34%	47.57%
	3	**55.25%**	46.11%	44.86%	43.74%	49.08%
	4	**57.13%**	46.79%	45.17%	43.89%	51.19%
Occasional	5-10	58.52%	**62.37%**	45.18%	44.75%	45.12%
	11-20	55.78%	**61.09%**	57.68%	44.12%	46.38%
Frequent	21-40	46.32%	52.89%	**59.79%**	53.13%	42.12%
	>40	34.86%	52.63%	**54.67%**	34.24%	31.86%

performs best for **Rare** customers, IBCF performs best for **Occasional** customers, and Implicit-ALS outperforms other methods for **Frequent** customers.

TABLE XI
Test results when considering the category of the recommended products.

	F1@10	DCG@10
ARM	15.26%	71.64%
IBCF	**17.87%**	**79.39%**
Implicit-ALS	16.39%	75.72%
BPRMF	12.61%	64.35%
MostPop	10.21%	58.48%

G. *Integrating Feedback from Marketing*

Acceptance of the system is difficult by the marketing department when precision is less than 20% since in our case this amounts to an average of 2 products among the 10 recommendations. After several fruitful discussions with the marketing department at TOTAL, the product managers pointed out that some products such as *Lays chips* and *Kettle chips*, can be treated as too similar. Therefore, from a marketing point of view, we should consider some recommendations at the category level, when estimating the performances of the algorithms.

To this end, we mapped each recommended product i to its corresponding category p_i. For example, if we recommend *Evian 1L*, we map it to the category *Water* and the same mapping is done for products appearing in the test set. This mapping can be qualified as optimistic since not all products belonging to the same category should be treated as similar to each other. However, this allows us to perform a first evaluation of our recommendation algorithms at the category level. Table XI shows the results when taking into account categories of products. Obviously, a significant improvement is achieved by all algorithms. We also note that IBCF remains superior. In the future, we will handcraft more refined taxonomies in collaboration with our partners at TOTAL and verify our algorithms' performance using those taxonomies. We expect however that the general trend we observed this time will remain the same.

VI. Related Work

Today, recommendations are implemented using various data mining and machine learning techniques. Popular approaches include content-based filtering [7] which compares the user's personal profile with the content of items that are to be recommended. These methods work well in practice when rich content is available. Another popular family of approaches is collaborative filtering [6], [28], from which we have selected various algorithms to design our experiments and case studies on our real-world dataset. Those approaches have the advantage to rely solely on the user-item interaction matrix with either explicit feedback such as ratings or implicit feedback such as purchases.

There are a few case studies that are closely related to ours, where authors designed comparative evaluations of different families of recommendation algorithms on customers' transactional datasets. A study by Huang et al. [29] points to the need for a better understanding of relative strengths and weaknesses of different types of algorithms, especially in e-commerce. Another study [30] showed that incorporating recommendation systems increased sales by up to 3.6%. They compared several recommendation algorithms on a Mobile Internet Application store, where they found that item-based

CF leads to a better accuracy when considering the number of clicks as the performance measure.

A more recent study [14] experimentally evaluated various CF algorithms on a dataset coming from a French building supplies chain. The simple bigram rules recommendation was found to yield the best accuracy. One important conclusion was that the relative performances of algorithms depend on the setting. Two settings were studied: complete and reduced purchase history, where the reduced one contains the event history of the most recent two weeks of the training data. In our paper, we adopted a more general setting where we evaluate algorithms with various training history lengths and found that a 6-month history yields the best performance. Similar observations were reported on other datasets and application domains [15] , [16], [17]. We also show that our findings are different from the literature since item-based CF performs best in our case. Additionally, we examine how our algorithms perform when considering the customers' purchase frequency, and show interesting results for different customer segments.

VII. Conclusion

In this applied research paper, we presented the results of a large-scale study of collaborative filtering recommendation algorithms on retail datasets. Our various results provide several insights to the marketing department of our partners at TOTAL. In particular, we are currently designing promotional campaigns based that leverage our results in different temporal contexts (morning, afternoon, evening and night), different regions and for different customer segments based on their purchase activity. The planned launch date of the promotional offers is December 2020 for a period of 3 months.

We would like to pursue two directions in the near future: a research direction for the design of promotional offers that combine customers' utility with business goals, and an experimental direction that studies the effect of the new algorithms on customer satisfaction and on revenue.

References

[1] A. S. Das, M. Datar, A. Garg, and S. Rajaram, "Google news personalization: scalable online collaborative filtering," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 271–280.

[2] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston *et al.*, "The youtube video recommendation system," in *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 2010, pp. 293–296.

[3] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, no. 1, pp. 76–80, 2003.

[4] F. Ricci, L. Rokach, and B. Shapira, "Recommender systems: introduction and challenges," in *Recommender systems handbook*. Springer, 2015, pp. 1–34.

[5] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 426–434.

[6] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The adaptive web*. Springer, 2007, pp. 291–324.

[7] M. J. Pazzani and D. Billsus, "Content-based recommendation systems," in *The adaptive web*. Springer, 2007, pp. 325–341.

[8] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *2008 Eighth IEEE International Conference on Data Mining*. Ieee, 2008, pp. 263–272.

[9] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-class collaborative filtering," in *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008, pp. 502–511.

[10] B. Sarwar, G. Karypis, J. Konstan, J. Riedl *et al.*, "Analysis of recommendation algorithms for e-commerce," in *EC*, 2000, pp. 158–167.

[11] C. Kim and J. Kim, "A recommendation algorithm using multi-level association rules," in *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*. IEEE, 2003, pp. 524–527.

[12] B. M. Sarwar, G. Karypis, J. A. Konstan, J. Riedl *et al.*, "Item-based collaborative filtering recommendation algorithms." *Www*, vol. 1, pp. 285–295, 2001.

[13] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 2009, pp. 452–461.

[14] B. Pradel, S. Sean, J. Delporte, S. Guérif, C. Rouveirol, N. Usunier, F. Fogelman-Soulié, and F. Dufau-Joel, "A case study in a recommender system based on purchase data," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 377–385.

[15] Y. Ding, X. Li, and M. E. Orlowska, "Recency-based collaborative filtering," in *Proceedings of the 17th Australasian Database Conference-Volume 49*. Australian Computer Society, Inc., 2006, pp. 99–107.

[16] S. Larrain, C. Trattner, D. Parra, E. Graells-Garrido, and K. Nørvåg, "Good times bad times: A study on recency effects in collaborative filtering for social tagging," in *Proceedings of the 9th ACM Conference on Recommender Systems*. ACM, 2015, pp. 269–272.

[17] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann, "Time-aware point-of-interest recommendation," in *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2013, pp. 363–372.

[18] G. Guo, H. Qiu, Z. Tan, Y. Liu, J. Ma, and X. Wang, "Resolving data sparsity by multi-type auxiliary implicit feedback for recommender systems," *Knowledge-Based Systems*, vol. 138, pp. 202–207, 2017.

[19] R. Agrawal, R. Srikant *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, vol. 1215, 1994, pp. 487–499.

[20] R. M. Bell and Y. Koren, "Lessons from the netflix prize challenge." *SiGKDD Explorations*, vol. 9, no. 2, pp. 75–79, 2007.

[21] Y. Koren, "The bellkor solution to the netflix grand prize," *Netflix prize documentation*, vol. 81, no. 2009, pp. 1–10, 2009.

[22] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, no. 8, pp. 30–37, 2009.

[23] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen *et al.*, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1235–1241, 2016.

[24] L. Lerche and D. Jannach, "Using graded implicit feedback for bayesian personalized ranking," in *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 2014, pp. 353–356.

[25] W. Pan, H. Zhong, C. Xu, and Z. Ming, "Adaptive bayesian personalized ranking for heterogeneous implicit feedbacks," *Knowledge-Based Systems*, vol. 73, pp. 173–180, 2015.

[26] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender systems handbook*. Springer, 2011, pp. 217–253.

[27] E. Brynjolfsson, Y. J. Hu, and M. D. Smith, "From niches to riches: Anatomy of the long tail," *Sloan Management Review*, vol. 47, no. 4, pp. 67–71, 2006.

[28] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, 2009.

[29] Z. Huang, D. Zeng, and H. Chen, "A comparative study of recommendation algorithms in e-commerce applications," *IEEE Intelligent Systems*, vol. 22, no. 5, pp. 68–78, 2007.

[30] D. Jannach and K. Hegelich, "A case study on the effectiveness of recommendations in the mobile internet," in *Proceedings of the third ACM conference on Recommender systems*. ACM, 2009, pp. 205–208.