World Wide Web (2020) 23:1497–1517
https://doi.org/10.1007/s11280-019-00738-1

Recommender system for marketing optimization

Wei Deng[1] · Yong Shi[1,2,3,4] · Zhengxin Chen[1] · Wikil Kwak[5] · Huimin Tang[1,6]

Received: 11 September 2018 / Revised: 27 April 2019 / Accepted: 23 September 2019 /
Published online: 18 November 2019

Abstract
Most of existing e-commerce recommender systems have been designed to recommend the right products to users, based on the history of previous users' individual transaction records. The real application scenarios of recommendation also have different requirements. From the customer point of view, many users visit the websites anonymously, so a practical way to provide anonymous recommendation is needed. From the marketing point of view, the recommendation list is not only a place to display the correlation of products, but also a place to display the variety of products as well as a tool to promote products. From the data point of view, concentration bias may be a serious problem. In this paper we propose trigger and triggered (TT) model to address all of these issues. First, the proposed model generates trigger and triggered pairs with significant correlations which can be used either to create a practical anonymous recommendation or as an input for products lifecycle modeling. The generated pairs not only reflect the relationships between products but also solve the problem of concentration bias very well. Besides, exposure of products required by marketing can be accomplished in the modeling. Second, by using the pairwise knowledge from the first step, the proposed model can recommend the right product at the right time to stimulate future consumptions and increase customers' engagement for the off-site case. A real-life retail store data is used to evaluate the proposed model, and the experimental results show that the model can decrease the problem of concentration bias while improving the correlation between recommendation items. The TT model significantly improves the sequential purchases on triggered items.

Keywords Recommender system · Concentration bias · Sequential purchase · Marketing optimization

✉ Yong Shi
yshi@unomaha.edu

✉ Huimin Tang
3120150654@bit.edu.cn

Extended author information available on the last page of the article

1 Introduction

The exponential growth of data changes the way business making decisions. The traditional marketing driven by human experiences shows its weakness: even though human decision matches with marketing goals at the company level, their observations on aggregated information cannot reflect the individual preference. Recommender systems help accomplish the marketing goals by presenting items to the users on the basis of personal interests as well as correlations between products. It stimulates more consumption due to the variety of products it can show. By feeding historical data, recommender systems predict the "rating" or "preference" score that a user would like to give for each item. The most common approach for a recommender system is collaborative filtering (CF) [1, 2], which makes predictions about a user's interest based on a collection of their previous consumption records and records from other users who have demonstrated similar patterns. For CF, there are two approaches: The neighborhood/memory-based and model-based approaches. The neighborhood approach is to choose items like the one that were previously purchased/viewed (item-based) or to recommend items that have been purchased by others with similar preferences (user-based). The model-based approach is to firstly construct an underlying model of user preferences, from where the predictions are inferred. The mainstream model-based CF algorithms include Bayesian Networks, Matrix Factorization, and Latent Semantic Models [3, 4]. Among them, the Factorization based method is most famous due to the Netflix movie recommendation competition with 1 million reward [5] and the winning algorithm Singular Value Decomposition (SVD) runs by transforming both users and items to the factor space with higher dimensions. Additionally, Koren's SVD++ [6] has further improvised it with implicit feedback information.

Even though existing recommendation systems achieve good accuracy through backtracking test, considering accuracy alone may lead to the phenomenon of "filter bubbles" [7], that isolates consumers from a diversity of content. Even worse, the accuracy inferred from historical records may not reflect the real correlations among products very well when "concentration bias" is heavy [8]. Besides, e-commerce recommendation needs to comply with the promotions in marketing, which means the content from recommendation in some cases is required not only to follow statistical trend, but also stimulate the sales of promoted products.

In this paper, we propose a new method called trigger and triggered (TT) model. It aims to solve the problems described above and provide a lifecycle recommendation. The proposed method consists of two parts. The first part eliminates concentration noise in the training data and can be used as independent anonymous recommendation system. The second part serves as a recommendation system with lifecycle awareness among products. The main contributions of this paper include the following aspects:

First, a pairs generation method is proposed to solve the problem of concentration bias on recommendations. We have analyzed sales data from a company which sells a wide range of products: not only furniture, but also electronics, appliances, floorings, toys and other living products as well. The proposed method can recommend diverse candidates other than only popular products.

Second, an anonymous recommendation is proposed for the on-site add-on recommendation with marketing optimization. This paper uses information at both category and item levels. The categorical connections between trigger-triggered pairs (*tt*-pairs) are generated based on trigger-triggered scores (*tt*-scores) and expert configurations. The right product pairs are found through linear programming where marketing goals can be added as constraints.

Third, a pair tracking algorithm is designed to encourage future consumption. The *tt*-pairs are pre-defined, but the timeframe between two products is various among different products as well as different customers. This paper uses the Weibull model [9] with three parameters to predict the timeframe between two products.

A real-life retail store data is used to evaluate the proposed model, and the experimental results show that the model can decrease the problem of concentration bias while improving the correlation between recommendation items. The TT model significantly improves the sequential purchases on triggered items.

The rest of this paper is organized as follows: Section 2 introduces terminologies and related techniques; Section 3 describes the proposed model; Section 4 introduces the data and gives the experimental results; Section 5 presents the conclusions and future direction of work.

2 Terminologies and related techniques

2.1 Score matrix

For convenience of discussion, this paper uses the term "score-matrix" to describe the score between different stock keeping units (SKU, a unique identification of each products). The score of SKU can be the number of times of co-purchases, or that of sequential purchases (a customer buys B because of the product A is bought first) or the similarity of products.

2.2 Weibull distribution

The Weibull distribution [9] is one of the most widely used lifetime distributions. It was originally proposed by the Swedish physicist Waloddi Weibull, who used the model to approximate the distribution of breaking strength of materials. The versatility of distribution can take on the characteristics of other types of distributions, by tuning value of the shape parameter α [9]. The probability density function of a Weibull model with a random variable x is shown in (1):

$$f(x) = \frac{\alpha}{\beta}\left(\frac{x-\mu}{\beta}\right)^{\alpha-1} e^{-\left(\frac{x-\mu}{\beta}\right)^{\alpha}} \tag{1}$$

where α, β and μ are known as the shape, scale and threshold parameters, respectively with constraints that $\alpha > 0$, $\beta > 0$, $\mu > 0$. The advantage of Weibull distribution is its versatility of distribution. Therefore, in this paper we use the characteristics to simulate distributions with a left long tail or right long tail, that looks like the sequential consumptions from a customer.

2.3 The methods of modeling and parameter tuning

2.3.1 Linear programming

Linear programming (LP) [10] is a method to find the optimal values of variables for a linear objective function. The object that the LP optimizes is a linear function, with some other linear equality or inequality constraints. LP's feasible region is a convex polytope, which is a bunch of sets defined by linear inequalities. The LP algorithm works to find the point or line in the polyhedron where generates the smallest value of the function.

The problem of LP can be expressed in canonical form as:

$$\begin{gathered} maximize c^{\mathrm{T}} x \\ subject to A x \leq b, x \geq 0 \end{gathered} \tag{2}$$

where x is the vector of variables, c and b are coefficients in vector format. A is coefficients in a matrix format.

2.3.2 Gradient descent

Gradient descent [11] is an optimization algorithm. It uses the method of first-order iterative gradient optimization to find the minimum of a function. To find a local minimum of a function, small steps are taken forward from the negative direction of the gradient of the function at current points. Stochastic gradient descent is a stochastic approximation of the gradient descent by aggregating a batch of gradient descent of sample data, which can largely increase the speed of parameter tuning [12]. Conjugate gradient descent derives from gradient descent, but instead of using gradient descent, it applies conjugate directions in the process of optimization [13].

2.4 Loss function and measurement

There are several metrics in the evaluation of recommender system [14, 15]: accuracy, coverage and diversity. Accuracy is the most important metric in a recommender system. A recommender system for top-k will give top k items to users in a sequence. There are multiple ways to measure its accuracy, for examlpe:

$$\boldsymbol{precision@k} = \frac{\left|T_{\boldsymbol{clicked}} \cap T_{K,\boldsymbol{recommended}}\right|}{K} \tag{3}$$

where $T_{\boldsymbol{clicked}}$ is the items have been clicked in the test set for a user, $T_{K,\ \boldsymbol{recommended}}$ is the k items recommended to a user.

If the rank of recommendation items is the major concern, the metric ap @ k will be used.

$$ap@k = \sum_{n=1}^{k} P(n)/min(m,k) \tag{4}$$

where $p(n)$ denotes the precision at the nth item in the item list.

For ap @ k metric, the same recommendation with different ranks will give different evaluations. For example, if user bought 3 items, follows recommended item #1 and #3, then $ap@10 = (1/1 + 2/3)/3 \simeq 0.56$. For the same recommendation list, if user follows item #1 and #10, then $ap@10 = (1/1 + 2/10)/3 = 0.4$.

The metrics are applied to evaluate the difference between estimated and actual purchase time, which are $\hat{y}_{\mathrm{u}}^{\mathrm{c}}$ and $y_{\mathrm{u}}^{\mathrm{c}}$. This paper uses the mean absolute error (MAE), the root mean square error (RMSE) and the mean absolute percentage error (MAPE) to evaluate the overall effect. The definitions are shown in (5), (6) and (7).

$$MAE = \frac{1}{N_u \times N_c} \sum_c \sum_u \left| y_u^c - y_u^c \right| \tag{5}$$

$$\boldsymbol{RMSE} = \sqrt{\frac{\sum_c \sum_u \left(y_u^c - y_u^c \right)^2}{N_u \times N_c}} \tag{6}$$

$$\boldsymbol{MAPE} = \frac{1}{N_u \times N_c} \sum_1^{N_u \times N_c} \left| \frac{y_u^c - y_u^c}{y_u^c} \right| \tag{7}$$

3 Trigger and triggered model

The goal of recommender systems is not only to satisfy customers but also to meet the demands of marketing. From the view of marketing, the accuracy along with the history data cannot be the only measurement, the goal of marketing is the other important metric. In addition, the other contribution in this method is to effectively connect experts' knowledge with algorithm. Unlike traditional recommender systems, Trigger and Triggered (TT) model concerns more on concentration elimination, marketing optimization and lifecycle of trigger and triggered products. The workflow moves from product to personalized granularity through two independent algorithms: TT_PAR and TT_PPE. TT_PAR is responsible for the generation of meaningful trigger and triggered pairs, and TT_PPE is for the lifecycle of sequential purchases. The relationship between these two algorithms is shown in Fig. 1.

The algorithm TT_PAR handles eliminating concentration noise and maximizing marketing goals quite well. Trigger and triggered pairs happened in short timeframe can be treated directly as an anonymous add-on recommendation, and positive results are shown in the experiment. The TT_PPE algorithm takes trigger and triggered pairs from each customer as inputs. By combining with customers' activities and demographic information, the lifecycles of trigger and triggered products are estimated, which can be used for both onsite and offsite

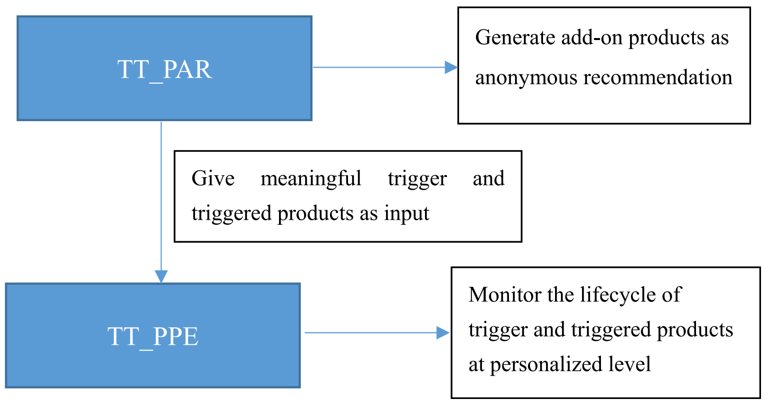
TT_PAR
Generate add-on products as anonymous recommendation
Give meaningful trigger and triggered products as input
TT_PPE
Monitor the lifecycle of trigger and triggered products at personalized level

Fig. 1 The workflow of TT_PAR and TT_PPE algorithms

recommendation and promotions. The TT_PPE algorithm provides more accurate prediction on lifecycle of product pairs by comparing with other practical algorithms.

3.1 Meaningful trigger and triggered pairs

When an algorithm is designed to auto recommend products, accuracy is always the most important criterion. However, focusing on accuracy alone may lead to the phenomenon of "filter bubbles", which means hot products become more popular but other options may decrease their exposure to customers as a result of sale diversity diminishing. For example, if a laptop is the most popular item in an e-commerce site and the popularity reaches to a critical point, the laptop might be treated as first recommendation option from the view of accuracy, no matter whatever the customer bought last time. This problem is even more serious when a store is featured by a certain type of products.

On the other hand, from the marketing perspective, accuracy is the primary objective. For instance, when the sales of refrigerator inexplicably decline, the recommender system should transform its role as a promotor to locate potential customers and increasing exposure of refrigerators.

Therefore, extracting meaningful trigger and triggered pairs in the history is important. For example, in a dataset, "iPhone - iPhone case" and "iPhone – bed" could be two pairs with the same numbers of co-purchase records. To deliver qualified results, the meaningless pair "iPhone – bed" should be excluded. This section illustrates the proposed method to run pair cleaning. The cleaned pairs can be used independently as an anonymous recommendation or combining with other information for lifecycle prediction.

3.1.1 Transformation of trigger and triggered pairs

Trigger-Triggered (TT) pairs are product pairs of a user (at categorical level) purchased at a different time. An example of an original transaction record for user u is shown in Table 1.

In a store, a customer normally buys multiple products at the same day, and the same product items or new items will appear in future consumption. As mentioned above, "the same product" purchase is defined as a repetitive purchase and "new item" purchase is defined as a complementary purchase.

In general, the *tt*-pairs are created between purchases at a different time. For example, the purchases at date 2018-01-02 and 2018-01-03 in the above table can generate two pairs shown in Table 2:

Table 1 The example of a customer's consumption records

Date	Product id
2018-01-02	1
2018-01-02	2
2018-01-03	1
2018-01-10	3
2018-01-10	1
2018-01-10	4
2018-01-13	3
2018-01-20	3

Table 2 The example of *TT*-pairs

tt-pair	time interval (days)
1–1	1
2–1	1

The same *tt*-pair can appear more than one time for a customer. The numbers of occurrence and relative time intervals are recorded as the format:

{user_id: [{*tt*-pair: [occurrences, [t1, t2, t3]]},...]}

where {} denotes dictionary structure with key-value and [] denotes list structure.

There are two exceptional cases which are used to exclude meaningless pairs:

Case 1: When a purchased product *i* is followed by several same purchases *j*, choose the most recent purchase *j* to create *tt*-pair. Assume that the future purchases *j* cannot be motivated by *i* if *i* has already triggered a purchase *j*. However, a future purchase *j* can be affected by new purchase *i* if it appears before *j*. For example, suppose the original purchase history for customer *u* is shown in Table 3.

If case 1 is applied, the following records in Table 4 cannot be used to create *tt*-pair since product 1 has already been used to trigger product 3 at date 2018-01-03. However, product 1 at date 2018-01-04 can be treated as a trigger for product 3 at 2018-01-05.

The original purchase history can be transformed to *tt*-pairs: {user_id: {*tt*-pair; occurrences; [t1, t2, t3]}} = {*u*: [{1–1: [1, [1]]}, {1–3: [2, [1, 2]]}, {3–3: [1, [3]]}]}. There are two reasons to apply this rule. Use "phone and phone case" as an example, for any user, if a phone has been purchased, the model predicts the time the user is going to buy the phone case and it can recommend at the most appropriate time. Therefore, first case in the purchase history can be used to generate "phone and phone case" pair for the training, while the other phone case consumptions will be treated as repetitive purchases of phone case instead of "phone and phone case" pair. The new "phone and phone case" pair for a user will be reactivated once a new phone is bought.

Case 2: When a purchased item *i* is followed by the other *i* and then purchased item *j*, the first *i* will be abandoned for *i-j* pair. For example, the original dataset for user *u* is shown in Table 5.

Table 3 The example of a customer's consumption records

date	product
2018-01-02	1
2018-01-03	3
2018-01-04	1
2018-01-06	3

Table 4 The example of records cannot be used to create *TT*-pair

date	product
2018-01-02	1
2018-01-06	3

If case 2 is applied, then the following records in Table 6 cannot be used to create *tt*-pair because the influence of product 1 at date 2018-01-02 is replaced by the other product 1 at date 2018-01-03.

The original dataset above can be transformed to *tt*-pairs: {user_id: {*tt*-pair; occurrences; [t1, t2, t3]}} = {*u*: [{1–1: [1, [1]]}, {1–3: [1, [2]]}]}. If we use “phone and phone case” as an example again, it is reasonable to create “phone and phone case” pair by using second phone purchase.

3.1.2 Extract meaningful pairs

In a dataset containing 10 million real-life records generated from 0.6 million customers from a US retail company, about 100 million *tt*-pairs were derived, and then aggregated to 1 million distinct *tt*-pairs at categorical level. The value of Gini coefficient [16] shows 71.9% on all purchase history demonstrates big concentration bias exists in original pairs. It is necessary to design an algorithm to decrease the bias in triggered items. A TT-Paris filtering method is proposed to solve the problem. It starts with a reverted ranking approach, and applies second-order mining (a post-stage of data mining projects in which humans collectively make judgments on data mining models’ performance.) [17, 18] to find meaningful pairs that are most important to the marketing. The *tt*-pairs filtering consists of two steps.

```
Algorithm 1 TT-Pairs Filtering

Step1: obtain reverted ranking using formula (8)

Step2: second-order mining by experts
```

Below we explain these two steps in detail.

Step 1: Rank the *tt*-pairs via *tt*-score, and extract *k* pairs with the highest grades.

The purpose of *tt*-score is to diminish the concentration bias in triggereds. *TT*-

Table 5 The example of a customer’s consumption records

date	product
2018-01-02	1
2018-01-03	1
2018-01-04	3

Table 6 The example of records cannot be used to create *TT*-pair

date	product
2018-01-02	1
2018-01-04	3

pairs are ranked by *tt*-score instead of numbers of occurrence. The *tt*-score is calculated as follows:

$$tt_{score} = nor\left(\left[O_{trigger_1 triggered_j}, O_{trigger_2 triggered_j}, \cdots, O_{trigger_n triggered_j}\right]\right) \qquad (8)$$
$$nor(\cdot) = \frac{x - x_{\min}}{x_{\max} - x_{\min} + \delta}$$

where O denotes occurrences and δ is a constant value used to increase weights of a list with higher occurrences. For instance, there are two lists: [1–3] and [10, 20, 30]; if δ is set to be 2, the first list is transformed to be [0, 0.25, 0.5] and the normalization of the second list is [0, 0.45, 0.9]. The purpose of δ is to give frequent pairs a relatively higher weight.

The following 6 products in Table 7 make three groups of pair:

If the topmost *tt*-pair from each group is kept, the results are "iPhone-bed", "carpet-bed", and "32 LED TV-bed" based on the co-purchase. However, the selected pairs cannot reflect the logical relationship among products. Alternately, *tt*-score normalizes occurrences dependent on each triggered. The triggered bed has three different triggers which form a list of three elements: [1000, 1500, 500]. After normalization it turns out to be: [0.5, 1, 0]. The normalized *tt*-pairs are shown in Table 8.

At this time the topmost *tt*-pair changes to "iPhone-case", "carpet-bed", and "32 LEDTV-iPad". The influence of concertation is eliminated. It is worth to note that data filtering is processed before applying *tt*-score, pairs with occurrences in the lowest 30% are deleted. δ is a constant value and is set to 300, which is used to balance the importance between *tt*-score and popularity. It is to ensure that popular pairs will get a higher score if other conditions are the same.

Table 7 The example of trigger and triggered pairs

Trigger	riggered	Occurrence
iPhone	bed	1000
iPhone	case	500
iPhone	iPad	600
carpet	bed	1500
carpet	case	400
carpet	iPad	300
32 LED TV	bed	500
32 LED TV	case	100
32 LED TV	iPad	400

Table 8 The example of trigger and triggered pairs after normalization

Trigger	Triggered	*tt*-score
iPhone	bed	0.5
iPhone	case	1
iPhone	iPad	1
carpet	bed	1
carpet	case	0.75
carpet	iPad	0
32 LED TV	bed	0
32 LED TV	case	0
32 LED TV	iPad	0.33

Figure 2 shows the comparison of *tt*-score with occurrence score among different triggered products when the trigger is a fitness accessory. It is found they are significantly different. More results are shown in the experimental section.

Step 2: Experts' opinions about the best products.

To collect experts' opinions about the important products, a filtering and grading system is designed for marketing experts. The experts are required to filter the most important pairs and to grade the products.

The filtering system is provided as a web service. An example is shown in Fig. 3:

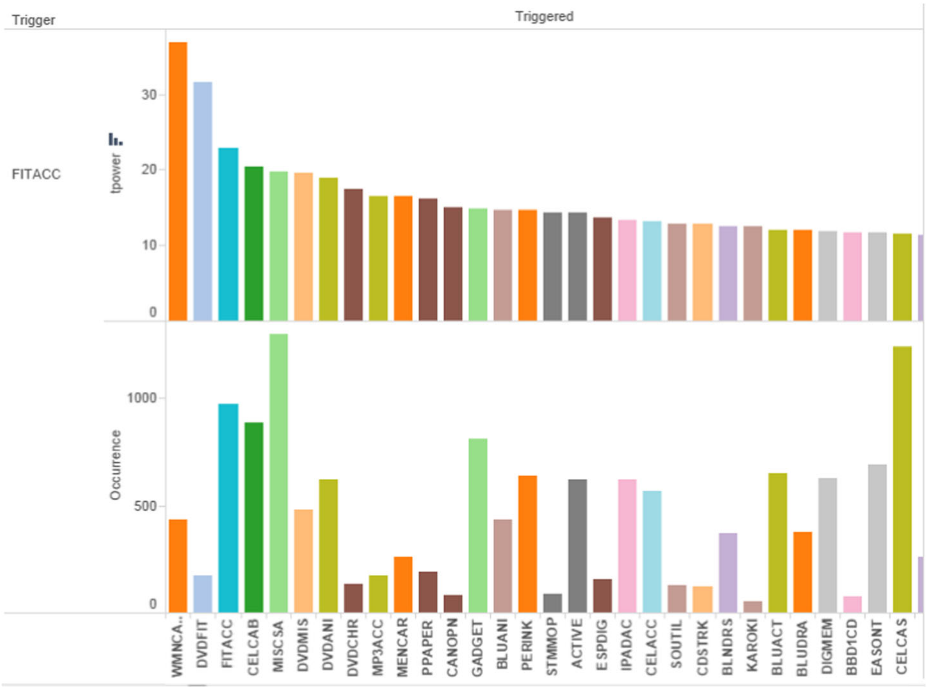

Fig. 2 The *tt*-score and the occurrence of each different triggered when the trigger is a fitness accessory

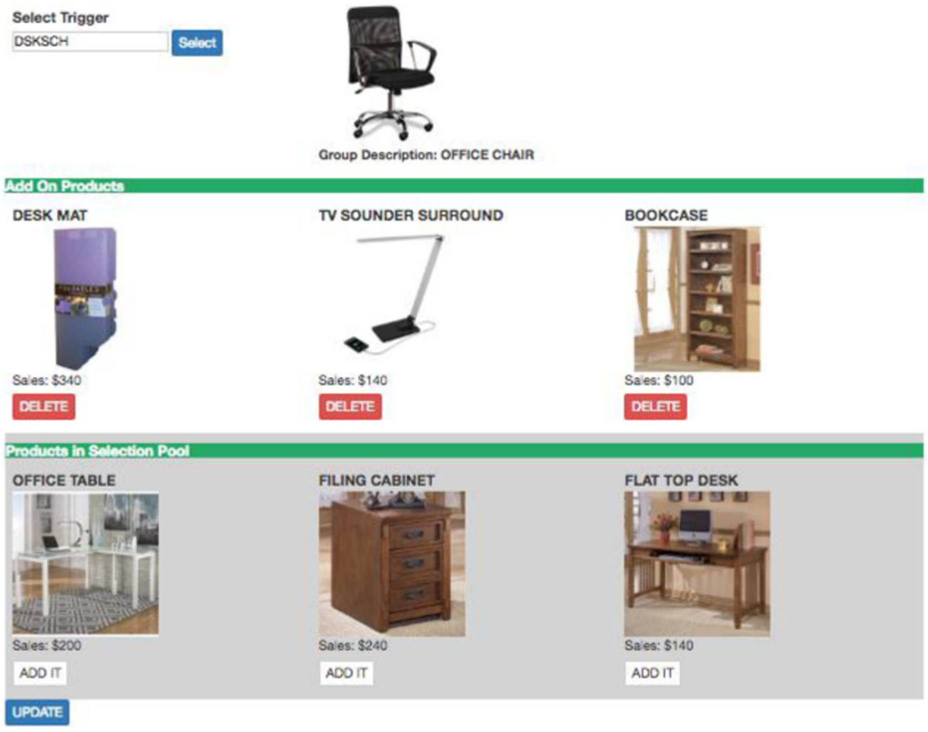

Fig. 3 The filtering system of *tt*-pairs

In this system, once a category code is selected, its triggered candidates are shown in the following two sections: “Add On Products” and “Products in Selection Pool”. The first section is what the expert chooses as the most important triggereds and the second section includes all candidates that can be added to the selected section.

For grading system, the marketing experts will give their weights to three components: margin, quantity and price for each product (in categorical level), and observe the correctness of their grades. The final scores for products are published after their adjustment.

The formula of calculating score is shown in (9):

$$p_{score} = \alpha \times \boldsymbol{margin} + \beta \times \boldsymbol{quantity} + \gamma \times \boldsymbol{price} \tag{9}$$

The product score shows the importance of products. Products with p_{score} higher than a threshold will be selected. Finally, part of the top k *tt*-pairs extracted from step 1 will be excluded if their p_{score} is less than the expert-defined threshold.

3.2 Trigger-triggered model for the anonymous recommendation

The anonymous recommendation should match three basic requirements: (1) reduce concentration bias in the dataset and reflect the logic correlation between products. (2) help the marketing to promote products. (3) guarantee the variety of products can be shown on the site.

To realize these goals, we use both the information of the category and SKU level of products. The categorical connections between *tt*-pairs are generated based on *tt*-scores and experts' selection which is described in section 3.1. The application of *tt*-scores and experts' inputs significantly reduce the problem of concentration bias. At the product level, linear programming is applied to find the right SKU pairs.

In Fig. 4, the matrix is the correlation score of products (*tt*-pairs). The size of the matrix is constrained by the generated group pair, and only relative SKUs in the groups can be connected. The scores in the matrix derive from the historical transactions. The formula of the score is shown in (10):

$$\boldsymbol{ppScore} = CO\text{–}\boldsymbol{Occurrence} + \alpha \times \mathrm{p}_{\boldsymbol{score}} \tag{10}$$

where $CO-\boldsymbol{Occurrence}$ is the number of co-occurrence of these two products in the last one year. The definition of co-occurrence is that the trigger and triggered products have been bought at the same time or the triggered products have been bought within 30 days after trigger products bought firstly. The linear programming problem (shown in (11)) is aimed to choose the best scores among all pairs while satisfying the goals of marketing promotion and recommendation variety will be reached.

$$\begin{gathered} maximize\ Z = \boldsymbol{ppScore}_{ij} \times \beta_{ij} \\ subject\ to \begin{cases} \beta_{ij} = 0 or 1 \\ \sum_{i=1}^{n} \beta_{ij}^{c} \le \varphi_j \\ \sum_{j=1}^{n} \beta_{ij}^{c} \le \omega_i \\ \sum_{js\ \boldsymbol{is\ marketing\ promtion}} \beta_{ij}^{c} = \delta_j \end{cases} \end{gathered} \tag{11}$$

where φ and ω are both a constant non-negative integer value, and δ is a constant positive integer value. The parameter φ is used to constrain the maximum numbers of products in each triggered category, that improves the variety of recommendation. The parameter ω is used to constrain the maximum numbers of a product can be shown as a triggered recommendation in the recommendation list. The value restricts over-recommendation on popular items. The parameter δ is used to promote specific products, makes sure the products can be shown more frequent at recommendation list.

		Group1		Group2		
		SKU1	SKU2	SKU1	SKU2	SKU3
Group1	SKU2	3	4	3	none	none
	SKU1	3	5	3	none	none
Group2	SKU3	4	5	none	2	none
	SKU2	5	none	none	5	none
	SKU1	3	5	none	5	none

Fig. 4 Scoring matrix

3.3 Trigger-triggered model for product promotion

Managing the lifecycle of consumption is a key to the success of customer engagement. Appropriate advertising should be sent to customers at the appropriate time in order to stimulate customers' potential consumption. For example, a good promotion plan should send customers a new iPhone promotion one or two year after the consumption of an iPhone, or a case promotion should be sent much earlier once the customer bought a phone. A product trigger and triggered system has been designed to track lifecycle of products at individual level, which can be used for product promotion by real time recommendation system or digital advertising through email or ad platforms. The principle of the system design is that once a certain product has been purchased, then the tracking system is activated, and related products will be sent to customers at right time if the products have not been purchased yet at that time.

To study the time frame between two sequential purchase activities in personalized level, it is to estimate the probability of a user's consumption at a time interval $(y, y + \Delta t)$. That is the conditional probability p(T∈$(y, y + \Delta t|$*product_u_i*): if the consumer u is going to buy product i in the future, what is the most likely time the consumer will buy. By examining the dataset, it is shown that the probability distribution of p(T∈$(y, y + \Delta t|$*product_u_i*) is usually a long tail with left or right peak. Therefore, a normal distribution may not be a good candidate to simulate it. Alternatively, a Weibull distribution has been applied. Wang et al. [19] have applied Weibull distribution to predict the time frame in ecommerce application. Different from that work, in the paper we use a Weibull model with three parameters to predict the time frame. Threshold parameter is included to deal with the case that some triggered items normally are not purchased immediately after the consumption of trigger products. In addition, we use gradient descent approach to tune the parameters instead of variational inference proposed in [19]. Even though gradient descent inference takes more time to locate a minimum, it will be easier to derive the algorithm. The probability density function of a Weibull model with random variable x is shown in (12).

As indicated in the formula, the scale parameter β is transformed to be a linear function of variables $\beta^T X$, where X is a vector of variables to capture signals of purchase, including a binary value indicating if the customer bought any same product or similar products in time bins t_1, $t_2 \ldots t_m$, or if triggereds have been purchased during promotion dates, or seasonality information, and etc. To make sure the derived scale parameter $\beta_1^T X > 0$, let's transform $\beta_1^T X$ to be $e^{\beta_1^T X}$. The derived density equation is:

$$f(y) = \frac{\alpha}{e^{\beta_1^T x}} \left(\frac{y-\mu}{e^{\beta_1^T x}} \right)^{\alpha-1} \exp\left(-\left(\frac{y-\mu}{e^{\beta_1^T x}} \right)^{\alpha} \right) \tag{12}$$

For each ith observation at each specific tt-pair c, the density function of purchase time is shown in (13):

$$p\left(y_i^c | X_i^c | \alpha^c | \beta^c | \mu^c\right) = \frac{\alpha^c}{e^{\beta_1^{cT} x^c}} \left(\frac{y_i^c - \mu^c}{e^{\beta_1^{cT} x^c}} \right)^{\alpha^c - 1} \exp\left(-\left(\frac{y_i^c - \mu^c}{e^{\beta_1^{cT} x^c}} \right)^{\alpha^c} \right) \tag{13}$$

The distributions of parameters are: $\alpha^c \sim N(\mu_\alpha, \delta_\alpha)$, $\mu^c \sim N(\mu_\mu, \delta_\mu)$, $\beta^c \sim N(\mu_\beta, \delta_\beta)$. It is denoted that $\omega = (\mu_a, \delta_a, \mu_\mu, \delta_\mu, \mu_\beta, \sum_\beta)$.

To solve the equation, we build a model separately for each product group m, where group refers to products at a particular categorical level. Therefore, in each group, the parameters are$\varphi = (\{\alpha^1, \mu^1, \beta^1\}, \{\alpha^2, \mu^2, \beta^2\}, \ldots, \{\alpha^m, \mu^m, \beta^m\})$. By grouping pairs, the purchase signal of similar products in a group is used. The joint likelihood for all variables is extended, as shown in (14):

$$L(\varphi|D_g) \propto L(D_g, \varphi) = p(\omega) \prod_{c=1}^{m} p(\alpha^c, \mu^c, \beta^c, |\omega) \prod_{i=1}^{n_m} p(y_i^c|\alpha^c, \mu^c, \beta^c, X_i^c) \tag{14}$$

Since the model contains many variables, the traditional method is computationally too expensive to get an answer. Instead, functions of parameters are replaced as constant c_i and MLE is used to estimate parameters, as shown in (15):

$$\begin{aligned}\varphi &= (\{\alpha^1, \mu^1, \beta^1\}, \{\alpha^2, \mu^2, \beta^2\}, \ldots, \{\alpha^m, \mu^m, \beta^m\}) = argmax\{L(Data, \varphi)\} \\ &= argmin\left\{\sum_{c=1}^{m}\left(c_1\|\alpha^c\|^2 + c_2\|\mu^c\|^2 + c_3\|\beta^c\|^2\right) + \sum_{c=1}^{m}\sum_{i=1}^{n_m} -log\left(p\left(y_i^c|X_i^c, \alpha^c, \beta^c, \mu^c\right)\right)\right\}\end{aligned} \tag{15}$$

The pseudocode to solve the previous equation is shown in the following algorithm 2. The parameters $(\mu_a, \mu_\mu, \mu_\beta)$ are initialized in the beginning. It is hidden parameters for grouping. Then Step 1 in the algorithm is to get a local minimal value of parameters φ, that is $(\{\alpha^1, \mu^1, \beta^1\}, \{\alpha^2, \mu^2, \beta^2\}, \ldots, \{\alpha^m, \mu^m, \beta^m\})$. On the basis of φ, parameters $(\mu_a, \mu_\mu, \mu_\beta)$ are updated. These two iteration steps continue until converge.

Algorithm 2 Trigger-Triggered Model

```
Initialize $(\mu_a, \mu_\mu, \mu_\beta) = (\mu_\alpha^0, \mu_\mu^0, \mu_\beta^0)$
$i = 0$
repeat
for tt-pair $= (1, 2, \ldots, m)$ in group:
    updating parameters $(\alpha^m, \mu^m, \beta^m)$ in (16) on the basis of known $(\mu_\alpha^i, \mu_\mu^i, \mu_\beta^i)$
end for
$i = i + 1$
updating parameters $(\mu_\alpha^i, \mu_\mu^i, \mu_\beta^i)$ in (17) based on known $(\alpha^m, \mu^m, \beta^m)$
end repeat (convergence)
```

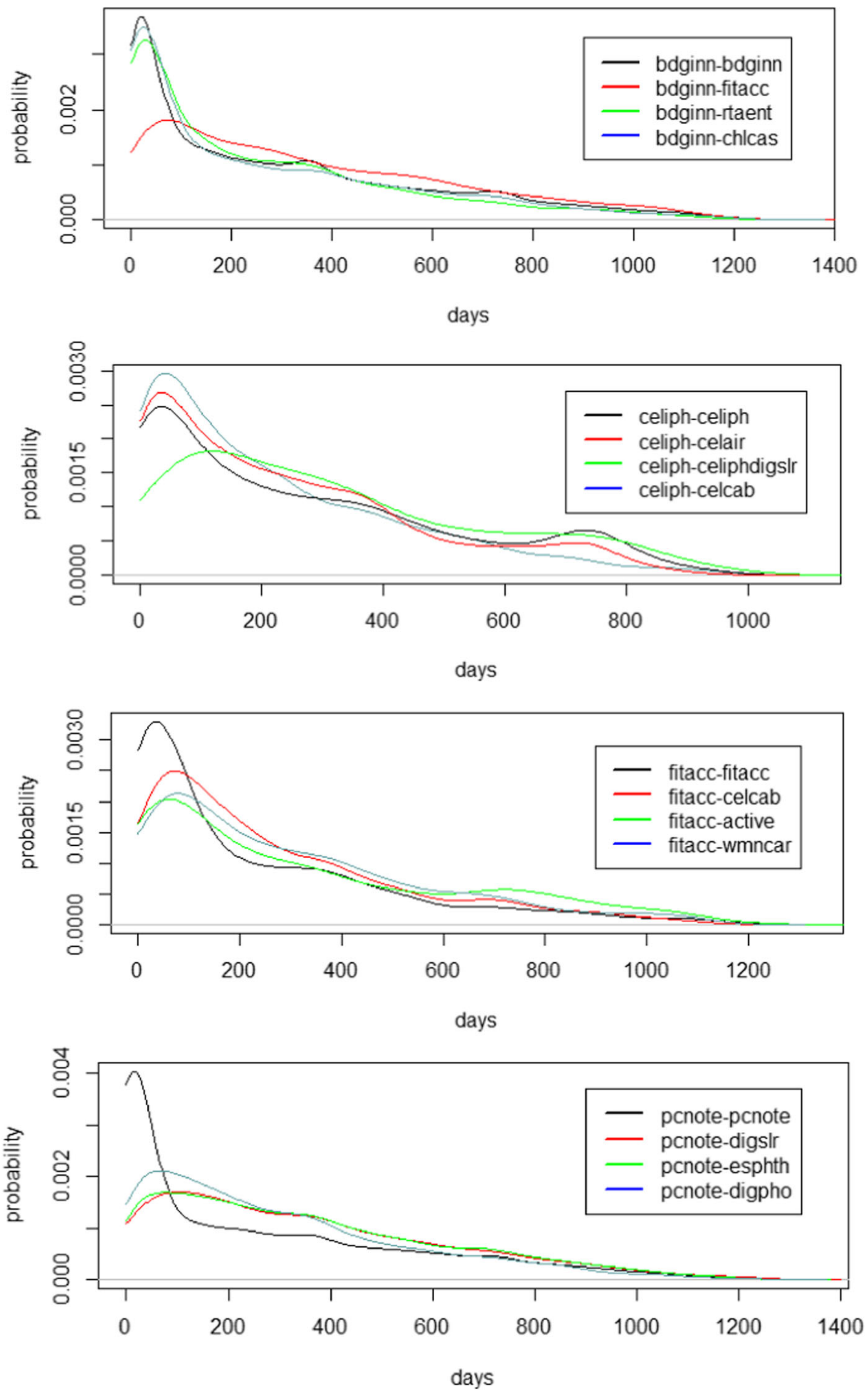

Fig. 5 The example of timeframes between trigger and triggered products

$$(\alpha^m, \mu^m, \beta^m) = \boldsymbol{argmin}\left\{c_1\|\alpha^c - \mu_a\|^2 + c_2\|\mu^c - \mu_\mu\|^2 + c_3\|\beta^c - \mu_\beta\|^2 + \sum_{i=1}^{n_m} -log\left(p\left(y_i^c | X_i^c \alpha^c, \beta^c, \mu^c\right)\right)\right\} \tag{16}$$

$$\left(\mu_a, \mu_\mu, \mu_\beta\right) = \boldsymbol{argmin}\left\{c_1\|\alpha^c - \mu_a\|^2 + c_2\|\mu^c - \mu_\mu\|^2 + c_3\|\beta^c - \mu_\beta\|^2 + c_4\|\mu_a\|^2 + c_5\|\mu_\mu\|^2 + \|\mu_\beta\|^2 \sum_{i=1}^{n_m} -log\left(p\left(y_i^c | X_i^c \alpha^c, \beta^c, \mu^c\right)\right)\right\} \tag{17}$$

Formulas (16) and (17) are hierarchical expression of (15) where prior knowledge are applied, two steps of inference improve the stability of prediction.

The updating methods at here can be algorithms such as Conjugate Gradient Descent, Broyden-Fletcher-Goldfarb-Shanno or others [11]. The optimum function [20] in R has been applied for parameters inference.

4 Experiments and results

4.1 Dataset cleaning and transformation

The transaction dataset is collected from a retail store including its online and in-store sales. There are more than ten million records in the data source. To make sure that pairs are meaningful and strong statistical signal can be presented, products can be aggregated into a categorical level. Data dictionary includes customer profile, in-store credit, categorical information and the description of products.

Data cleaning is a critical step to decrease noise. Infrequent users who have transaction records less than three times a year have been deleted. Most frequent users such as commercial customers (buyers from entrepreneurs) have been filtered out as well. In addition, sale-and-return pairs have been carefully matched and deleted. Split orders (order at the same day but delivery at a different time. In the dataset, it is presented as different order) have been merged together and pinned to a unique order date.

Table 9 The accuracy test of *tt*-pairs

Timeframe between purchases	Top X recommendation	Percentage of hit (%)	Average times of hit (%)	Percentage of hit for random guess (%)	How much better did the model perform?
< 30 days	5	32	42	0.464	70X
< 30 days	10	41	61	0.93	45X
< 30 days	15	47	74	1.4	34X
< 30 days	20	50	82	1.8	28X
31–60 days	5	30	43	0.464	70X
31–60 days	10	40	64	0.93	45X
31–60 days	15	45	76	1.4	34X
31–60 days	20	48	86	1.8	28X
61–90 days	5	28	44	0.464	70X
61–90 days	10	37	64	0.93	45X
61–90 days	15	42	75	1.4	34X
61–90 days	20	44	83	1.8	28X

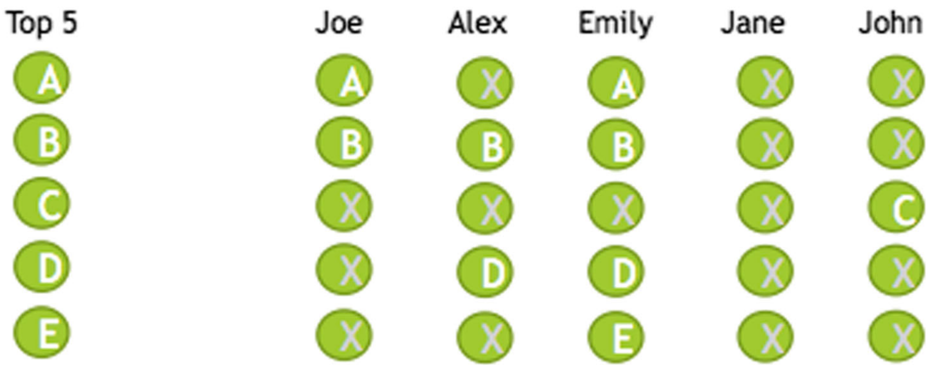

Fig. 6 An example of customers' purchase

Once the original dataset goes through the *tt*-pair extractor, the dataset will be processed as the format <y; *tt*-pair; customer_id>. Here y is a time frame in day granularity; *tt*-pair means the trigger and triggered pair modeled, and customer_id tells us who generates this consumption sequence. The other dataset <customer_id; *tt*-pair; ×1, ×2,…,xn > includes the value of attribute vector for each customer_id and *tt*-pair, which will be fed as x in (13).

The timeframes of *tt*-pairs are showed in Fig. 5, which follows a long-tail distribution. It matches the distribution of Weibull distribution:

bdginn, caliph and etc. are the codes of specific products, the pairs among these codes indicate the distribution of timeframes of sequential purchase over all customers.

4.2 The results of trigger and triggered pairs extraction

In the experiments, the verification of extraction rule for *tt*-pairs is the first step. In order to test, extraction rule in section 3.1 is applied to the entire transaction dataset from 2012 to 01-01 to 2015-01-01. For each trigger product, its top 5, 10, 15 and 20 triggered products are calculated within timeframes 1–30 days, 31–60 days and 61–90 days. Based on all these rules, A total of 1000 customers are randomly selected, these customers have at least one more follow-up purchase within 1–30 or 31–60 or 61–90 days after their last purchase in the training dataset

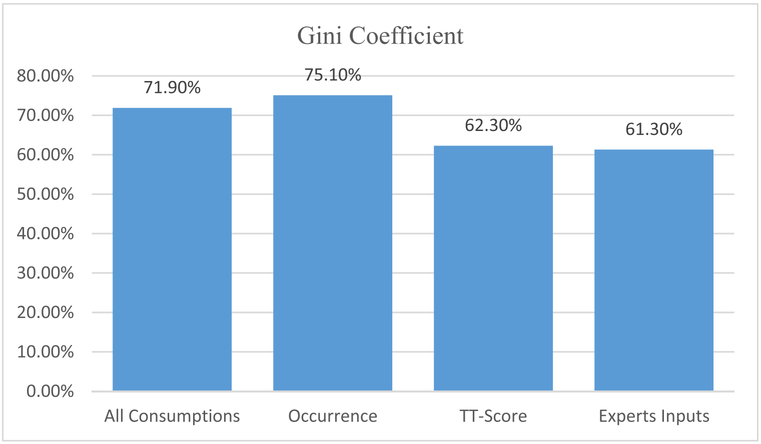

Fig. 7 The diversity comparison of Gini coefficient

(2012-01-01 to 2015-01-01). The products recommended by *tt*-pairs are compared with customers' actual purchase. The result is shown in Table 9.

In Table 9, "Timeframe between purchases" is the time frame between last purchase in the training dataset and following new purchase. "Top X recommendation" is numbers of recommendation pairs from *tt*-pairs. "Percentage of hit" shows the percentage of customers have purchased at least 1 of the top k. "Average hit" indicates how many of the total purchases the model has predicted correctly. The meaning of the two metrics "Percentage of hit" and "Average hit" is illustrated in Fig. 6:

Suppose all five persons have bought product X, and the *tt*-pairs system recommends A, B, C, D, E as their next purchase in the next 30 days. After comparing with customers' real purchase records, it is shown that everyone except Jane purchased at least 1 of the top 5 so "Percentage of hit" is 80%. Since 9 of the 25 products is correct, so "Average hit" is 36%.

"Percentage of hit for random guess" in Table 9 is the random selection of 5,10,15,20 products from the product pool for comparison purpose. The last column in Table 9 is the comparison of *tt*-pairs with random guess. It illustrates that *tt*-pairs significantly reflect customers' purchase sequences. Therefore, further modeling for pairs is worthy.

Gini coefficient [9] is applied to test the diversity of recommended contents. It is also used to measure the inequality among values of a frequency distribution as a percentage between 0 and 100, G = 0% indicates that all of the frequencies are the same whereas G = 100% means absolute difference. Mathematically Gini coefficient can be expressed as:

$$G = 1-2\int_0^1 L(u)du \tag{18}$$

where $L(u)$ is the Lorenz curve.

To test, all top 5 recommended triggereds/items from each trigger are collected. There are three ways to list the top 5 recommended triggereds. The first is to rank by the numbers of occurrences in the *tt*-pairs database; the second is to rank by the *tt*-score in the *tt*-pairs database; the third is to rank after experts' filtering. The test is applied to evaluate the Gini coefficient of all triggered items from all triggers. The result is shown in Fig. 7:

The Gini coefficient of recommendation based on the original frequency is 75.1%, even higher than 71.9%. It is easy to see that recommendation based only on frequency has a severe problem of concentration bias. After the clean of the *tt*-score algorithm, the Gini coefficient decreases to 62.30%. Experts' inputs after *tt*-score do not impact Gini Coefficient too much. It should be noted that the goal of "second order" mining is not to decrease Gini coefficient.

4.3 Results of trigger-triggered model for product anonymous recommendation (tt_par)

Since TT_PAR has not been published to the real system yet, and there is no data to verify the accuracy of the model. The effect of TT_PAR has been tested by marketing experts. The recommendation results from TT_PAR have been compared with the results from Coremetrics

Table 10 The accuracy comparison

Method	ap @ 3	precision @ 3	ap @ 5	precision @ 5
TT_PAR	0.81	0.86	0.76	0.82
Coremetrics	0.49	0.54	0.42	0.52

Table 11 The accuracy comparison

Error	Mean Value	Median Value	Linear Reg.	TT_PPE_init	DNN Reg.	TT_PPE
MAE	32.80	35.32	25.21	23.12	22.70	18.20
RMSE	47.06	52.12	36.11	34.93	34.28	30.89
MAPE	1.91	2.10	1.35	1.17	1.29	1.03

[21], the experts chose the recommended items which they feel appropriate. The result is shown in Table 10:

The results of TT_PAR are significantly better than the results of Coremetrics. From the recommendation contents of Coremetrics, it is easy to figure out the severe problem of concentration bias. TT_PAR performs much better due to the step of TT pair cleaning, and the concentration-avoid constraints in the modeling.

4.4 Results of trigger-triggered model for product promotion extraction (tt_ppe)

To test the effects of TT_PPE, we extract customers in transaction dataset from 2012-01-01 to 2015–01-01who both have trigger and triggered items at different times. The difference between two dates are y, and only the records that time frame y is less than 120 days are kept. There are two ways to calculate prediction value y, one is the mode of Weibull distribution [22], the problem of mode formula is the restriction on shape parameter, which is required to be larger than 1. The other alternative we applied in the paper is the method proposed by Wang et al. [19], it is defined as:

$$q(y,x) = Pr(y < T \leq y + \Delta t | T > y) = \frac{Pr(T \leq y + \Delta t) - Pr(T \leq y)}{1 - Pr(T \leq y)} \tag{19}$$

The TT_PPE model has been compared with mean value, median value, linear regression, TT_PPE_init, and deep neural network (DNN) regression [23]. The DNN uses 5 hidden layers with ReLU activations, and Adam is used as the optimizer. Errors between estimated and actual purchase time are shown in Table 11 and Table 12.

TT_PPE_init is a simple version of TT_PPE without hierarchical inference, which means all follow-up purchases in different categories use the same model and share parameters. TT_PPE fits one Weibull distribution per product category. The results show that the TT model is much better than linear regression, Mean value and Median value estimation. Even though TT_PPE_init is not as good as DNN regressor, the TT_PPE is better than DNN regression after inferencing by using categorical information. The results prove that the nature of different categorical consumption is not the same, and TT_PPE benefits from it.

Table 12 The accuracy comparison

Error (double Purchase)	Mean Value	Median Value	Linear Reg.	TT_PPE_init	DNN Reg.	TT_PPE
MAE	41.67	44.50	34.52	33.36	29.89	27.84
RMSE	54.72	56.90	45.18	42.39	40.10	39.53
MAPE	2.93	2.63	2.30	2.21	1.87	1.92

5 Conclusion and future works

In this paper, we have investigated the following issue: how to make e-commerce recommendation on concentration biased dataset, and how to optimize marketing goals on both personal and anonymous recommendations. The way to eliminate concentration bias has been answered by a method of trigger-triggered pairs extraction; Anonymous recommendation with marketing goals has been answered by the model TT_PAR and personalize recommendation for promotions has been answered by the model TT_PPE.

TT_PAR uses the method of linear optimization. The variety of recommendation and correlation among products can be achieved by two-level constraints and the minimum marketing exposure can be reached by the other constraint. TT_PPE implements promotion recommendation through the Weibull model and the importance of time factor is considered. Experiments with real-life data have shown the effectiveness of models.

However, in this study, the models are built only on historical transactional data, the online real-time behaviors from customers are not included. Therefore, further studies will focus on how to leverage the information from online streaming data to improve the click and conversion ratio in recommendation systems.

Funding This work was partially supported by grants from National Natural Science Foundation of China (No. 91546201, No. 71331005, No. 7193000078).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Pennock, D. M., Horvitz, E., Lawrence, S., Giles, C. L.: Collaborative filtering by personality diagnosis: a hybrid memory- and model-based approach. Sixteenth Conference on Uncertainty in Artificial Intelligence, 473–480, (2000)
2. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. International Conference on World Wide Web, 285–295 (2001)
3. Yang, X., Guo, Y., Liu, Y.: Bayesian-inference-based recommendation in online social networks. IEEE Trans. Parallel Distrib. Syst. **24**(4), 642–651 (2013)
4. Hofmann, T.: Latent semantic models for collaborative filtering. ACM Trans. Inf. Syst. **22**(1), 89–115 (2004)
5. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. Eighth IEEE International Conference on Data Mining, 263–272, (2009)
6. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining: 426–434 (2008)
7. Nguyen, T. T., Hui, P. M., Harper, F. M., Terveen, L., Konstan, J. A.: Exploring the filter bubble: The effect of using recommender systems on content diversity. Proceedings of the 23rd international conference on World wide web (2014)
8. Adamopoulos, P., Alexander, T.: On over-specialization and concentration bias of recommendations: Probabilistic neighborhood selection in collaborative filtering systems. Proceedings of the 8th ACM Conference on Recommender systems (2014)
9. Iii, J.E.P., Wiener, J.G., Smith, M.H.: The Weibull distribution: a new method of summarizing survivorship data. Ecology. **59**(1), 175–179 (1978)

10. Solow, D.: Linear and Nonlinear Programming. Wiley Encyclopedia of Computer Science and Engineering (2007)
11. Shanno, D.F.: On broyden-fletcher-goldfarb-shanno method. J. Optim. Theory Appl. **46**(1), 87–94 (1985)
12. Bottou, L.: Large-scale machine learning with stochastic gradient descent: 177–186, (2010)
13. Hager, W.W., Zhang, and Hongchao: A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM J. Optim. **16**(1), 170–192 (2005)
14. Celma, Ò., Herrera, P.: A new approach to evaluating novel recommendations. 13(8), 179–186 (2008)
15. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inform. Syst. **22**(1), 5–53 (2004)
16. Fleder, D. M., Hosanagar, K.: Recommender systems and their impact on sales diversity. ACM Conference on Electronic Commerce: 192–199 (2007)
17. Zhang, L., Li, J., Li, A., Zhang, P., Nie, G., Shi, Y.: A new research field: intelligent knowledge management. International Conference on Business Intelligence and Financial Engineering, 450–454, (2009)
18. Nie, G., Zhang, L., Zhang, Y., Deng, W., Shi, Y.: Find intelligent knowledge by second-order mining: three cases from China. IEEE International Conference on Data Mining Workshops, 1189–1195 (2010)
19. Wang, J., Zhang, Y.: Opportunity model for e-commerce recommendation: right product; right time. International ACM SIGIR Conference on Research and Development in Information Retrieval, 303–312 (2013)
20. Nash, J. C., Varadhan, R., Grothendieck, G., Nash, M. J. C., Yes, L.: Package ‘optimr’, (2016)
21. Davenport, T. H.: What do we talk about when we talk about analytics. Enterprise analytics, optimize performance, process and decision through big data, 9–18, (2013)
22. Nassar, M.M., Eissa, F.H.: On the Exponentiated Weibull distribution. Commun. Stat. **32**(7), 1317–1336 (2003)
23. Aurélien, A. G.: Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. O'Reilly Media Inc (2017)

Affiliations

Wei Deng [1] • Yong Shi [1,2,3,4] • Zhengxin Chen [1] • Wikil Kwak [5] • Huimin Tang [1,6]

[1] College of Information Science and Technology, University of Nebraska at Omaha, Omaha, NE 68182, USA

[2] Research Center on Fictitious Economy and Data Science, The Chinese Academy of Sciences, Beijing 100190, China

[3] Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences, Beijing 100190, China

[4] School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190, China

[5] College of Business Administration, University of Nebraska at Omaha, Omaha, NE 68182, USA

[6] School of Management and Economics, Beijing Institute of Technology, Beijing 100081, China