Next-Basket Recommendation Constrained by Total Cost

Brainstorming session report

Oleg Lashinin[1], Marina Ananyeva[1,2]

[1]Tinkoff, Moscow, Russia

[2]National Research University Higher School of Economics, Moscow, Russia

Abstract

The next-basket recommendations are a well-studied problem in the academic field. As a rule, we want to predict top-K goods that might be added to the last known basket of the user based on the previous purchase history. However, the existing studies do not take into account that each user can be limited by the amount of money. Therefore, clients of online websites might be interested in the opportunity to receive personal recommendations with a given fixed cost. For example, we can take the average spending over the history of customer's purchases as an upper bound. In this article, we consider addressing the problem of making next-basket recommendations with set total cost restrictions.

Keywords

next-basket recommendation, constrained recommendations

1. Motivation

The majority of online platforms and e-commerce stores feature recommender systems that consider users' previous purchases and try to predict their future purchases by solving the next-basket recommendation task.

The evaluation process of the next-basket recommendations implies that the considered method generates recommendations of some predefined length K. Lists of recommendations are compared to the last basket in a test sample by computing the ranking metrics, e.g. such as *Recall@K* and $NDCG@K$. These lists have a fixed length as far as e-commerce marketplaces offer millions of items, which makes it almost impossible to view all of them. Moreover, most user interfaces, especially mobile applications, have very limited space to display item lists. Thus, it is crucial to maximize both the high precision and ranking of recommendations with a fixed K length.

Meanwhile, we could take into consideration another restriction for the next-basket formalization task. Each user usually has a certain amount of money that he or she can afford to spend at the moment of purchase.

The following might be one of the possible user interfaces for personal recommendations with total cost constraints. If the user specifically sets this limit, we provide a list with recommended items of some arbitrary length, not exceeding the indicated total cost. Furthermore, the client may specify such constraints in a real-time session by using filters or other UI tools and obtaining refreshed recommendations with a total sum up to the given constraint. Otherwise, the need to use constrained recommendations is evident for most online platforms even without updating the user interfaces. For example, the e-commerce site might recalculate the lists of recommendations for different online shopping cart states in real time. Once a client adds or deletes any good in a basket, we can update the block with recommendations under the basket taking into account the updated total sum. Depending on the business demands, the total cost can be formalized in a task as the maximum sum of the user's purchase or as the total cost of recommendations only.

In this extended abstract, we propose to use a new approach that helps users to view affordable recommended goods and e-commerce platforms to use the available slots for recommendations wisely, maximizing the probability of purchases.

2. Proposed approach

We formulate the problem of constrained next-basket recommendations as follows. Let there be U - a set of users and I - a set of items. Each user u has a list of predicted relevance scores for items $r_{u,i} = \{r_{u,1}, \dots, r_{u,|I|}\}$. It is implied, that any recommender model can be used to obtain these scores, which is beyond our scope. Also, we obtain a list of items prices $p_i = \{p_1, \dots, p_{|I|}\}$. For each user, we have a true basket with some purchased items in the last transaction $b_{u,j} = \{b_{u,1}, \dots, b_{u,|J|}\}$ where $|B|$ is a number of goods in the last purchased basket. The aim is to predict items in $b_{u,j}$ with a K-length recommendations $\hat{r}_{u,k} = \{\hat{r}_{u,1}, \dots, \hat{r}_{u,K}\}$. TC_u is a total cost for a user u. We generate the list based on the restrictions for each user:

ORSUM@ACM RecSys 2022: 5th Workshop on Online Recommender Systems and User Modeling, jointly with the 16th ACM Conference on Recommender Systems, September 23rd, 2022, Seattle, WA, USA

o.a.lashinin@tinkoff.ru (O. Lashinin); m.ananyeva@tinkoff.ai (M. Ananyeva)

1234-5678-9012 (M. Ananyeva)

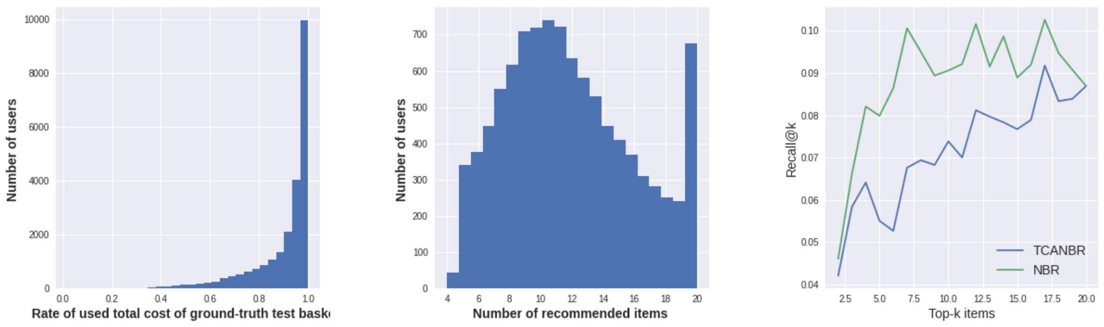

Figure 1: Results of experiments on the Ta-Feng dataset. From left to right, the first histogram demonstrates the average rate of the used total cost in recommended baskets. The second histogram presents the distribution of the number of recommended items. The line graph presents Recall@k w.r.t. different k values both for the classical NBR problem and the proposed next-basket with constraints task.

$$\sum_{i=1}^{K} r_{u,i} \rightarrow max \tag{1}$$

$$\sum_{i=1}^{K} p_{u,i} < TC_u \tag{2}$$

However, a few remarks are required here. Usually, we have no information on how many items $|B|$ a customer will add to the next cart. To overcome this problem, we can generate the lists $\hat{r}_{u,k}$ for different K values and select those that maximize the equation (1). In our implementation, we generate K-length vectors with recommendations for each user and drop extra items Alternatively, we can try to predict the number of items $|B|$ and make the lists based on the predicted values.

Also, the TC_u parameter can be varied by the user in real-time. The user can change the thresholds of the total cost and select the best option with personal recommendations.

We use a genetic algorithm in order to find the optimal $\hat{r}_{u,k}$. The generated populations of multi-hot vectors with N size from the most relevant items $r_{u,i} = \{r_{u,1}, \dots, r_{u,N}\}$ for each user. The i-th component of a generated vector (gene of a chromosome) is supposed to be equal to 1 and the corresponding i-th element of the recommendations list will be included to the final predictions. If the condition (2) is not met, the fitness function is equal to negative infinity or $\sum r_{u,k}$ otherwise. Therefore, the genetic algorithm will maximize (1) with the respect to the condition (2). We leave other possible approaches to finding the optimal solution for future work.

3. Experiments

Dataset. We conducte preliminary experiments on the [1, 2] Ta-Feng dataset[1], which is extensively used in recommender systems.

It includes purchase history of goods with given pricing. Users in the Ta-Feng dataset can purchase different quantities of the same goods. Nonetheless, we will consider proposing a product based on the dataset's average price. Furthermore, we assume that the recommendations cannot contain duplicates, which means that each item can only be suggested once and in a single occurrence. We employ the same data preparation strategy as in TIFU-KNN [1] to evaluate our approach, and we use a leave-one-basket [3] scenario.

Metrics. We compute the ranking metrics (Recall@k, NDCG@k) that are commonly used in the next-basket recommendation evaluation [1, 4, 2].

Experiment Settings.

In offline evaluation settings, the TC_u value can be chosen by the researchers. For instance, we could predict the total cost of the last known basket for each user or vary the values of TC_u across some range on a dataset. In particular, it is possible to calculate the average basket cost from the training set for each user and take this aggregated statistic as a constraint. In our case, TC_u equals the true cost of the last basket $\sum p_{g_{u,j}}$ of each customer. It approximates the case when the customer knows exactly how much he is can afford or is willing to spend.

We adopt the genetic algorithm, using the implementation from the open-source framework geneticalgorithm2[2]. For solving this task on the Ta-Feng dataset, we use only the 20 most relevant items and population

[1]https://www.kaggle.com/datasets/chiranjivdas09/ta-feng-grocery-dataset
[2]https://github.com/PasaOpasen/geneticalgorithm2

size of 40 vectors. PersonTopFreq approach is applied as a recommender model that returns the most popular items from the baskets of all users, demonstrating the high performance in TIFU-KNN [1]. Also, we use negative ranks of items as $r_{u,i}$ because negativity is needed for maximizing the TC_u of items included in the final predictions. Application of other recommendation approaches for the next basket task is left for future work.

Performance Comparison. Figure 1 (a) shows the average rate of the used total cost, which is quite close to the cost of the basket from the testing sample. In this graph, all predicted baskets with a higher total cost are neglected. The costs of most collected baskets are close to TC_u and satisfy the user's needs.

Figure 1 (b) demonstrates the number of goods that were included in the baskets, which is close to the Gaussian distribution.

Figure 1 (c) compares the values of Recall@k w.r.t. the different k values for two cases. The results when we recommend only top-k relevant items are in green. The Recall@k values with top-k items with the total cost restrictions from the 20 most relevant items are in blue. There is no extreme difference between the two methods observed, but our recommendations fit additional restrictions in the TCANBR scenario.

4. Discussion

During the discussion panel of this brainstorming ides, that took place at the 5th Workshop on Online Recommender Systems and User Modeling (Seattle, WA, USA, 2022)[3], we can summarize the open issues and several future directions for future work.

Additional constraints. Taking into account the formalization problem mentioned above, we could think of additional restrictions that can be applied to bring the solution closer to real conditions. Firstly, we could extend to the lower and upper bounds of the price for recommended items. It can either be set as hyperparameters or as extra constraints in our task. It would help to solve a problem of making the list of recommendations only with cheap or expensive items. Secondly, we could extend the task to introducing the profit margins, which both customers and enterprises pursue considering business metrics.

Bundle recommendations. Considering the fact that we want to generate a list of recommended items, we might think about it as bundle recommendations. In particular, it could enhance the diversity or complementarity of predicted items. Thus, the approaches for bundle recommendations could also bee used to solve the task of constrained recommendations with total cost. However, the questions on how to evaluate the bundle, attribute the reward, and the conditions for buying the entire bundle or only certain items from it are open. In addition, the different cases which allow a user to add or delete an item allow the platforms to make a user-friendly interface with updates in recommendations in real time. In particular, after each customer's action with the basket, the recommended bundle of items can be recalculated entirely or iteratively, by replacing each added or deleted item with a new item of the same price in order not to exceed the total price. Finally, we could use bundle recommendations in the offline experiments as baseline solutions or adapt them to this formalized task.

Approaches. Instead of genetic algorithms, we could also try to adopt a graph-based A* algorithm, integer programming, or knapsack problem with dynamic programming techniques to address the problem of generating next-basket recommendations with a total cost constraint. Still, some open issues should be discussed. For instance, should the sum of relevance scores be equal to the total relevance score and be constrained by the lower bound of relevance or not? What are the reasonable restrictions for relevance scores considering their different ranges of values, negative and positive scores from the recommender algorithms? These are only a few examples of open questions.

Total Cost values. In this paper, we suggested a way to set a total cost value. However, we could think of predicting the total cost for each user, for example, using linear regression or other machine learning models. In user cases, when the customer updates the maximum affordable cost, we can take this value in real-time and use it on the inference step. Otherwise, we could also think of using the segments of customers instead of personalized values of the total cost. For example, the customers with low, medium, and high income and three different values for this constraint correspondingly.

Item quantity. One of the problems, which we neglect for the simplicity, is the challenge to take into account the different quantities for each item in the lists of recommendations. In case we want to add this additional restriction to the task, we should think carefully about how to include this constraint and optimize the function. We also should think about whether we allow duplicated items. For example, the milk, but of different brands and prices. In addition, some items could be bought always in single pieces, but some others are taken in multiple pieces or are the goods sold by weight. In our case, we implied no duplicates and only 1 allowed piece of each item for simplicity.

The list of problems mentioned above is not exhaustive and can be continued and developed into comprehensive versions of formalization task and be addressed by different approaches.

[3]https://orsum.inesctec.pt/orsum2022/

5. Conclusion

In this article, we briefly introduced a formalization problem of the next-basket constrained recommendations problem. The tested approach simulates a case when a customer of an online marketplace can spend a particular upper bound of the amount of money. Alternatively, some online services can fit the recommendations into a suggested budget for each user. This problem task suggests generating item recommendations that do not exceed the specified maximum total cost.

References

[1] H. Hu, X. He, J. Gao, Z.-L. Zhang, Modeling personalized item frequency information for next-basket recommendation, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1071–1080.

[2] G. Chen, Z. Li, A new method combining pattern prediction and preference prediction for next basket recommendation, Entropy 23 (2021) 1430.

[3] Z. Meng, R. McCreadie, C. Macdonald, I. Ounis, Exploring data splitting strategies for the evaluation of recommendation models, in: Fourteenth ACM conference on recommender systems, 2020, pp. 681–686.

[4] M. Ariannezhad, S. Jullien, M. Li, M. Fang, S. Schelter, M. de Rijke, Recanet: A repeat consumption-aware neural network for next basket recommendation in grocery shopping (2022).