JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 18, No. 8, pp. 1561-1566, Dec. 2017

암묵적 피드백 기반 반려동물 용품 추천 시스템

최 희 열[1*] · 강 윤 희[2] · 강 명 주[3]
[1]한동대학교 전산전자공학부
[2]백석대학교 정보통신학부
[3]트리니티(주) 빅데이터사업부

Pet Shop Recommendation System based on Implicit Feedback

Heeyoul Choi[1*] · Yunhee Kang[2] · Myungju Kang[3]
[1]School of Computer Science and Electrical Engineering, Handong Global University, Pohang 37554, Korea
[2]Division of Information & Communication, Baekseok University, Cheonan, 330-704, Korea
[3]Division of BigData Business, Triniti, Inc., Seoul 07997, Korea

[요 약]

기계 학습과 인공 지능 기술의 발전으로 다양한 응용분야들이 가능해지고 있고, 이중에 추천 시스템은 이미 여러 업체들에서 영화 추천이나 상품 추천 등의 서비스에 적용하여 효과를 보고 있다. 이러한 서비스 중인 추천 시스템들의 대부분은 아이템의 내용을 분석하여 추천하거나 아니면 평점과 같은 직접적인 피드백에 기반하여 시스템을 학습하고 추천하고 있다. 하지만 많은 온라인 쇼핑몰 중에는 아이템의 내용을 분석하는 것이 어렵고, 직접적인 피드백 정보가 없거나 혹은 거의 없어 추천 시스템 구축이 어려운 경우가 많다. 이러한 경우에도 사용자의 상품 조회에 관한 로그 기록들은 어렵지 않게 확보할 수 있고, 로그 기록들만 가지고도 추천 서비스를 제공할 수 있다면 서비스의 질을 향상할 수 있을 것으로 기대된다. 본 논문에서는 사용자의 로그 기록으로부터 암묵적인 피드백인 상품 조회 정보를 추출하고, 암묵적인 피드백에 기반한 추천 시스템을 구현하고, 제안된 시스템은 온라인 반려동물 용품점에 적용하여 확인한다. 즉, 사용자들의 상품조회를 위한 클릭정보만을 활용하여 반려동물 용품 추천 시스템을 구축하여 서비스로 확인한다.

[Abstract]

Due to the advances in machine learning and artificial intelligence technologies, many new services have become available. Among such services, recommendation systems have already been successfully applied to commercial services and made profits as in online shopping malls. Most recommendation algorithms in commercial services are based on content analysis or explicit feedback rates as in movie recommendations. However, many online shopping malls have difficulties in content analysis or are lacking explicit feedbacks on their items, which results in no recommendation system for their items. Even for such service systems, user log data is easily available, and if recommendations are possible with such log data, the quality of their service can be improved. In this paper, we extract implicit feedback like click information for items from log data and provide a recommendation system based on the implicit feedback. The proposed system is applied to a real in-service online shopping mall.

색인어 : 기계 학습, 추천 시스템, 암묵적 피드백, 상품 클릭 정보
Key word : Machine learning, Recommendation systems, Implicit feedback, Click information on items

http://dx.doi.org/10.9728/dcs.2017.18.8.1561

Received 07 December 2017; **Revised** 23 December 2017
Accepted 25 December 2017

***Corresponding Author; Heeyoul Choi**

Tel: +82-54-260-1303
E-mail: hchoi@handong.edu

Ⅰ. Introduction

Recently, the advances in machine learning and artificial intelligence technologies have made many new services available including image caption generation and neural machine translation [1], [2], [3]. Among such services, recommendation systems have already been successfully applied to commercial services (e.g. Amazon and Netflix) and have made profits. Most recommendation algorithms in commercial services are based on content analysis (content-based filtering) or explicit feedback rates (collaborative filtering) to recommend a romantic partner or movie [4], [5]. Compared to content-based filtering methods, collaborative filtering methods are more popular because it can be applied to any domain application as long as a user-item matrix is available. The user-item matrix usually includes explicit feedback like the rates of items.

However, many online shopping malls have difficulties in content analysis or are lacking explicit feedbacks on their items, so recommendations are not available for their items. Even for such service systems, user log data is easily available and abundant including purchasing, search, and click information. Although such data does not represent user preference, if recommendation is possible with such log data, the quality of their service can be improved.

For collaborative filtering methods, most algorithms are based on matrix decomposition or factorization based on a user-item matrix with explicit feedback [4], [6]. Instead of using explicit feedback information, implicit feedback can be used and the algorithm needs to be changed since the implicit feedback is noisy and the preference is uncertain as opposed to explicit feedback [7].

In this paper, we extract implicit feedback like click information for items from log data, implement a recommendation algorithm based on the implicit feedback, and build a web-based recommendation system for a real in-service online shopping mall.

This paper is organized as follows. Section 2 gives the research background including collaborative filtering and collaborative filtering based on implicit feedback. Section 3 describes the proposed system and its implementation in Python. Experiment results are presented in section 4. Section 5 summarizes our work.

Ⅱ. Background

The two popular approaches for recommendation systems are content-based filtering and collaborative filtering. The former one is based on content like items and user profile descriptions. It recommends items similar to what the user likes. The collaborative filtering is based on a community which generates a user-item matrix with users' item preference. It recommends items that are popular among other users who have similar preference to the user.

Since our system is a collaborative filtering method based on implicit feedback, we briefly describe collaborative filtering especially with implicit feedback.

2-1 Collaborative filtering

Given a user-item matrix based on explicit preference, there are two basic techniques for collaborative filtering: the user-oriented neighborhood method and item-oriented neighborhood method. Those methods are used to estimate the preference of user u on item i. The user-oriented methods find neighbors who like the same items as the user in the past, and who have rated item i. Finally, the weighted average of the rates on item i from the neighbors is a prediction of how much the user will like item i. The item-oriented methods are used to find directly similar items to item i by considering the similarity of the item vectors where each element indicates the preference of users.

Instead of the basic techniques, recently, latent factor approaches like singular value decomposition (SVD) and matrix factorization are applied. Especially, nonnegative matrix factorization (NMF) is one of the popular matrix factorization methods [4]. Fig. 1. represents matrix factorization, and Eq. (1) shows the objective function that needs to be optimized.

$$\min_{p,q} \sum_{(u,i)\in K} (r_{ui} - p_u^T q_i)^2 + \lambda\left(\sum_u \| p_u \|^2 + \sum_i \| q_i \|^2\right), \quad (1)$$

In Eq. (1), K is the set of users and items with feedback, and r_{ui} is the known rating of user u for item i. The second term is a regularization term on p_u and q_i. After training we have p and q matrices, and simply $p_x^T q_y$ is the estimated rate for user x on item y. The dimension of p_u and q_i should be given.

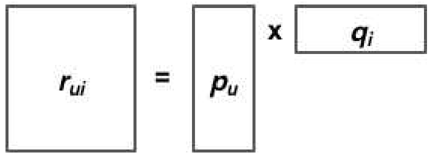

Fig. 1. Matrix factorization. r_{ui} is a user-item matrix where each element indicates a user's preference on the item.

2-2 Implicit feedback based collaborative filtering

While explicit feedback such as likeness rates on items represents users' preference on items, the meaning of implicit feedback such as click information depends on the context and need to be considered in different ways.

First, there is no negative implicit feedback, and it is not clear to estimate how much the user likes an item. For example, even when a user purchases an item, the user might not like his item. In other words, purchasing history does not include negative feedback like low rate on an item.

Second, when a user clicks on an item a lot, it does not mean that he or she likes it a lot. In other words, the implicit feedback value indicates a user's confidence about an item rather than preference as in explicit feedback. Actually, a user might click on an item repeatedly when she is not sure about it. That is, we cannot predict preference based on implicit feedback, rather we estimate whether the user would click on the item or not.

Finally, the objective function needs to reflect the characteristics of implicit feedback. We do not predict the number of clicks on items, instead we have to predict whether the user will click on a certain item or not.

Considering the difference of implicit feedback from explicit feedback, the objective function and the recommendation method are changed as in Eq. (2) [7].

$$\min_{p,q} \sum_{(u,i)\in K} c_{ui}(f_{ui} - p_u^T q_i)^2 + \lambda(\sum_u \| p_u \|^2 + \sum_i \| q_i \|^2)$$
, (2)

In Eq. (2), $c_{ui} = 1 + \alpha r_{ui}$ which is a confidence in observing f_{ui}. f_{ui} is 1 if r_{ui} is positive, otherwise 0. r_{ui} is the number of clicks or purchases rather than rates as in Eq. (1). The other parts are the same as the explicit feedback based collaborative filtering method.

Ⅲ. Proposed System

3-1 Feedback extraction

To reflect users' new activities, we have to retrain our recommendation system with new log data. To do so, we implement an FTP client to extract click information from log data as in Fig. 2. The FTP client makes a csv file in the recommendation server, and the csv file includes user id, item id, and click time.

3-2 Algorithm Implementation

Once a new csv file is ready, Rec Server restarts. Rec Server instantiates a recommend class and the class's constructor includes training the recommendation system with the new csv file. Including the recommendation functions as well as the constructor and training funciton, the class's member functions are summarized as follows:

- __init__() : It calls train function.
- preprocess(input, output): It changes the click information from the 'input' file into count information saved in the 'output' file.
- train(train_file, factor): It trains with 'train_file' which is the preprocessed file. The factor is the dimension of p_u and q_i. The default is 50.
- recommend_by_item(item, k=5): It recommends k items, when the user clicks an item. Output includes item list with the corresponding scores.
- recommend_by_user(user, k=5, avoid_cs): It recommends k items, when a user login. If 'avoid_cs' is 1, a new user will receive recommendations with the hottest items, to avoid the 'cold start' problem.

3-3 System architecture

The recommendation system consists of two subsystems: Rec Server and an FTP client. Those subsystems are written in Python 2.7. Rec Server gets a request for recommendations with an item id from the online shopping service when a web client selects the item. The Rec Server handles a request as a HTTP GET method from a web client, by generating a query for a recommendation with regard to the request and calling the recommend class member functions. Fig. 2 describes the overall recommendation system.

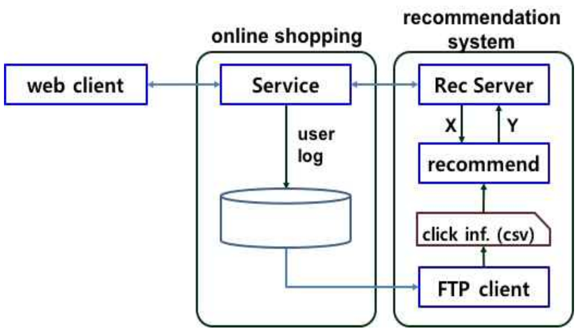

Fig. 2. An overview of the proposed recommendation system. It consists of an FTP client and Rec Server including the 'recommend' class.

Ⅳ. Results

4-1 Data

The target of the recommendation is only for dog and cat related items. The number of users and items are changing as the service runs. For a certain amount of time, the number of relevant users and items are 12,832 and 7,488, respectively, with 49,109 click events.

4-2 Recommendation results

It is not easy to evaluate recommendation systems, because offline data has not been affected by recommendations. Online evaluation can report sales increases, although it is hard to analyze the contribution to the increase by the recommendation system. In this paper, we simply report the offline test to show the exact performance of the proposed system. That is, we split the data: the first 90% of the data for training and the last 10% for testing. After the training process is converged, when 10 recommendations are predicted, 37% of the recommendations are clicked on by the users in the test data. Note that the users was not aware of the recommendation since it is an offline evaluation.

In addition, to provide the performance of the proposed system, we provide actual recommendation results and explain how it is used in service. Since the dog and cat items are almost completely separated in the log data, we train two recommendation systems, one for dog items and the other for cat items. Table 1 and 2 represent two example results with user id, item id and likelihood scores for new items for the users.

Table 1. Actual recommendation for dog items.

user id	item id	likelihood scores
19648	470	0.464096313311
	2569	0.453683529038
	7481	0.448605660653
	538	0.445067602371
	7076	0.373438927726
19636	2705	0.4800791339
	2759	0.419951944636
	10167	0.411738725819
	8487	0.408034235901
	4362	0.401761217472

Table 2. Actual recommendation for cat items.

user id	item id	likelihood scores
19627	6483	0.141586075419
	5222	0.125568881776
	10821	0.0923567972193
	7157	0.0922985993294
	5249	0.083323005802
19856	8180	0.12130842182
	6918	0.11099323707
	9173	0.096495717576
	460	0.0955009754033
	437	0.0933257951198

Finally, we applied the proposed system to an online shopping service. The recommendation system works as follows: When a user logs in or clicks on an item, recommendations are given based on user id or the item id. The likelihood scores are presented to indicate how likely the user would click on the corresponding item which is calculated by the inner product of p_u and q_i obtained by Eq. (2). Note that the numbers are not users' preferences but the likelihood that the users would click the corresponding item, so we need further investigate how much the proposed recommendations are connected to the sales.

Since the pet shop does not have feedback information on items from users, so that explicit feedback based collaborative method cannot be considered. However, we proved that the proposed system could predict how likely users would click an item based on the implicit information extracted from user log data.

Ⅴ. Conclusion

When explicit feedback was not available, we extracted implicit feedback like click information on items from the log data, and implemented a recommendation system based on the implicit feedback. To apply the proposed system to an in-service online shopping mall, we built a web-based recommendation system including data transfer, training the model, and recommendation. The system is running in a real online shopping mall.

For future work, we need to investigate how much sales profit the recommendation makes. Also, recent advances in deep learning techniques especially recurrent neural networks can be applied.

Acknowledgement

This research was supported by No. 2170043 of Handong Global University Research Grants.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature,* Vol. 521, No. 7553, pp. 436–444, May 2015.

[2] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in *Proceeding of the International Conference on Learning Representations*, San Diego: CA, May 2015.

[3] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, “Show, attend and tell: Neural Image Caption Generation with Visual Attention,” in *Proceeding of the International Conference on Machine Learning*, Lille: France, pp. 2048-2057, July 2015.

[4] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *IEEE Computer Society*, Vol. 42, No. 8, pp.30-37, August 2009.

[5] H. R. Choi, B. I. An, and G. I. Chung, “Mobile Context Based User Behavior Pattern Inference and Restaurant Recommendation Model,” *The Journal of Digital Contents Society*, Vol.18, No. 3, pp.535-542, June 2017.

[6] H. J. Kim and S. Y. Chung, “A Recommender System Using Factorization Machine,” *The Journal of Digital Contents Society*, Vol. 18, No. 4, pp.707-717, July 2017.

[7] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in *Proceeding of the International Conference on Data Mining*, Pisa: Italy, pp. 263-272, December 2008.

Heeyoul Choi

2005: Dept. of Computer Science and Engineering, POSTECH (M.S.)
2010: Dept. of Computer Science and Engineering, Texas A&M University (Ph.D)
2010 ~ 2011: Indiana University (PostDoc)
2015 ~ 2016: University of Montreal (Visiting Researcher)

1998~2001: OromInfo (Programmer)
2011~2016: Samsung Electronics, SAIT (Research Staff Member)
2016~present: Handong Global University (Assistant Professor)
※ Research Interests : Machine Learning, Deep Learning, Artificial Intelligence

Yunhee Kang

1993: Dept. of Computer Engineering, Dongguk University (MSE)
2002: Dept. of Computer Science, Korea University (Ph.D)
2003: Carnegie Mellon University (Visiting Researcher)
2010~ 2011: Indiana University (Visiting Researcher)

1991 ~ 1994: ETRI (Researcher)
1994 ~ 1997: KCAF (Senior Researcher)
1997 ~ 2000: Orom Information (Manager)
2000 ~ present: Baekseok University (Associate professor)
※Research Interests : Cloud computing, Big-data computing, Fault Tolerance

Myungju Kang

1988: Dept. of Computer Engineering, Dongguk University (BSE)
1991: Dept. of Computer Engineering, Dongguk University (MSE)

1993 ~ 1997: Jeju Industry University (Assistant professor)
2000 ~ 2008: Chief of Research, ClickQ, Inc.
2009 ~ 2012: Head of SI Business Department, Java Information Technology, Inc.
2013 ~ present: Division of Big Data Business, Triniti, Inc.
※Research Interests : Big Data, NLP, AI