Leverage Item Popularity and Recommendation Quality via Cost-sensitive Factorization Machines

Chih-Ming Chen[*†], Hsin-Ping Chen[*], Ming-Feng Tsai[*] and Yi-Hsuan Yang[†]
[*] Department of Computer Science, National Chengchi University, Taiwan
[†]Research Center for Information Technology Innovation, Academia Sinica, Taiwan
Email: cmchen@citi.sinica.edu.tw, s10019@cs.nccu.edu.tw, mftsai@cs.nccu.edu.tw, yang@citi.sinica.edu.tw

***Abstract*—The accuracy of recommendation trends to be worse towards the long tail of the popularity distribution of items, but items in the long tail are generally considered to be valuable as they occupy a majority part of entire data. In this paper, we develop an instance-level cost-sensitive Factorization Machine (FM) to tackle the problem. The new algorithm allows the FM model to automatically leverage the trade-off between item popularity and recommendation quality. Specifically, by adding a cost criterion to the loss function, the FM model is now able to discriminate the relative importance of popularity from massive data. In addition, we convert several well-known functions into the popularity weighting functions, thereby demonstrating that the proposed method can fit the model parameters to various kinds of measurements. In the experiments, we assess the performance on a real-world music dataset which is collected from an online music streaming service, KKBOX. The dataset contains 1,800,000 listening records that cover 5,000 users and 30,000 songs. The results show that, the proposed method not only keeps the performance as primitive model but also avoids retrieving too much popular music in the top recommendations.**

I. Introduction

Most existing algorithms for recommendation ignore unpopular or newly introduced items and focus on retrieving items that have already been associated with too many ratings. Although this strategy can get sensible performance, for real-world applications it is desirable to recommend unpopular items to the users. For instance, for music recommendation, as people might have listened to those popular songs before, it does not add too much value if a system recommends songs the user already knows. In consequence, we cannot always satisfy user needs by recommending popular items. The goal of music recommender system is to help users discover new music and recall the music users would like to listen to. In light of this observation, we seek to develop a mechanism that can avoid recommending too many popular songs at the top of the recommendation list.

Recommending too popular items is also referred to as the problem of lacking of novelty [1]. The main reason is that the frequently appearing items occupy the great part of listening records over the whole dataset, which is the so-called "long tail" phenomena. Some recent studies started to focus on how to deal with the phenomenon in recommendations [2]–[4]. In short, popular items are indeed relevant but easy to be out-of-dated. Consequently, many new techniques have been proposed to mitigate popularity bias in recommendations [5]–[8]. Most work approaches this by seeking a balance between the trading-off of accuracy and item popularity. In this paper, we propose to tackle this problem by introducing the idea of the cost-sensitive learning. In particular, given that Factorization Machines (FMs) have been shown superior to many competing methods for a variety of recommendation problems [9]–[13], we propose a novel instance-level cost-sensitive Factorization Machine algorithm in this paper. With the proposed algorithm, we can effectively reduce the popularity bias without hurting the quality of the recommendation.

Cost-sensitive learning is an learning technique that utilizes different penalties for different error situations [14]. The idea is simple yet powerful in handling an imbalanced dataset [15]–[17]. For instance, to alleviate the popularity bias, we can increase the costs of unpopular items. Cost-sensitive learning can be applied to various kinds of algorithms including probabilistic model [18], tree model [19], support vector regression for classification problem [20]. With the similar approach, we embed the weighting function in the loss function of FM model. Accordingly, the model is able to automatically leverage the recommendation accuracy and any specific features including the popularity.

We conduct our experiments on a music dataset which is collected from KKBOX (http://www.kkbox.com), a leading on-line music streaming service in Asia. Because of a partnership with this company, we are able to have access to the listening records of people who subscribe to the KKBOX service. From the listening records we also identify the long tail problem. In the experiments, two different weighting functions are used for weighting the item popularity. The experimental results show that, by using the unpopular music favor function, the average popularity of the recommendation list can be greatly reduced, especially in the top recommendations. Moreover, the reduction of popular items does not affect the recommendation quality in terms of recall.

The technical contributions of this paper include: 1) We propose a cost-sensitive FM that is able to obtain a better tradeoff between the recommendation accuracy and the popularity of recommended items. 2) Two weighting functions are proposed, and the weighting function can be directly embedded into the learning phase. 3) We provide experimental result validates that the proposed method is able to reduce the average popularity of recommendations and remain the recommendation quality. Although the present paper might be at best preliminary, it represents one of the first attempts that introduces the idea of cost-sensitive learning to the FM framework, to our best knowledge.

The paper is organized as follows. We present the FM algorithm and the proposed cost-sensitive FM in Section II.

978-1-4799-4274-9/14 $31.00 © 2014 IEEE
DOI 10.1109/ICDMW.2014.62

IEEE computer society

Then we describe the experiment setting and results in Section III. Finally, we conclude the paper in Section IV.

II. Methodology

To begin with, we introduce the standard FM model in the first, and then to give a detailed description for the idea of cost-sensitive FM afterward.

A. Standard FM

Factorization Machine is a generic framework that can simulate many successful matrix-factorization-based models via feature engineering. Given a design matrix $X \in \mathbb{R}^{n\times p}$ and its record tuple $(\mathbf{x}, y)$, the standard 2-way factorization machine is defined as:

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{j=1}^{p} w_j x_j + \sum_{j=1}^{p}\sum_{j'=j+1}^{p} \hat{w}_{jj'} x_j x_{j'}, \quad (1)$$

where w_0 models the global bias, w_j models the weight of j-th features x_j, and $\hat{w}_{jj'}$ models the weight of each pair of features. Since all features in FM model are transferred into indicator variables, the framework may suffer from the data sparsity problem. Rendle fuses the factorization technique with the model; that is, the $w_{jj'}$ is factorized into the pairs of interaction parameters [10]: $\hat{w}_{jj'} = \sum_{f=1}^{k} v_{j,f} v_{j',f}$. From technical perspective, k is a hyperparameter that determines the model complexity and allows the model to learn the latent factors within the features. In brief, FM provides the promising framework for the ranking problem. For more details, please refer to the [9].

B. Cost-Sensitive FM

The cost sensitive learning technique is to give different costs to each misclassified situation. Therefore the learning model can be sensitive to the high weighting errors. Similar ideas of using weighting functions with latent factor models can be found in the literature [21]–[23]. We apply this mechanism to the loss function of FM and make it feasible to leverage the recommendation accuracy and target instances. In this paper, we focus on recommending the relative unpopular music because such unpopular songs are sometimes more likely to satisfy the user need.

1) Cost-sensitive loss function: In the FM model, a typical optimization task usually involves with a loss function l over the observed data S:

$$OPT(S) = \underset{\Theta}{\operatorname{argmin}} \sum_{(\mathbf{x},y)\in S} l(\hat{y}(x|\Theta), y) + \sum_{\theta\in\Theta} \lambda_\theta \theta^2. \quad (2)$$

L2 regularization (e.g. λ) is applied to the function as well, the goal of which is to prevent the model from over-fitting problem. Specifically, Θ represents the model parameters and the function is used to calculate the difference between the prediction value $\hat{y}$ and true value y. In this paper, the least square loss is proposed as a cost weighting function: $l(\hat{y}, y) = (\hat{y} - y)^2$. We employ an intuitive way to extend the loss function to cost-sensitive version. That is, enhancing the loss effects by multiplying an additional cost:

$$l_c(\hat{y}, y) = c_{\mathbf{x}}(\hat{y} - y)^2. \quad (3)$$

We use $c_{\mathbf{x}}$ to denote the corresponding cost of instance $\mathbf{x}$. Based on the framework, the learning model trends to be fitted more on the high weighting instances.

2) Learning methods: There are three different ways to optimize the model parameters during the learning stage. We describe how to embedded the cost-sensitive loss function in each learning method below.

The first learning method is stochastic gradient descent (SGD) [10]. For SGD learning, the model parameters of FM can be updated by gradient descent methods:

$$\theta \leftarrow \theta - \eta\left(\frac{\partial}{\partial\theta} l_c(\hat{y}, y) + 2\lambda_\theta \theta\right), \quad (4)$$

where η is the learning rate controlling the learning steps.

The second learning method is alternating least-squares (ALS) [11]. ALS learning is another kind of approach that obtains the optimal value by minimizing the total loss per model parameter. The solution of regularized cost-sensitive loss function can be found by the first derivative of the optimization criterion.

$$\frac{\partial}{\partial\theta} OPT(S) = \sum_{(\mathbf{x},y)\in S} 2(\hat{y}(x) - y) h_\theta(x) c_{\mathbf{x}} + 2\lambda_\theta \theta. \quad (5)$$

Note that FM is a linear combination of two functions g_θ and h_θ: $\hat{y}(\mathbf{x}) = g_\theta(\mathbf{x}) + \theta h_\theta(\mathbf{x})$. Therefore the minimum value is obtained while the derivative equals to 0:

$$\theta^* = -\frac{\sum_{(\mathbf{x},y)\in S} (g_{(\theta)}(\mathbf{x}) - y) h_\theta(x) c_{\mathbf{x}}}{\sum_{(\mathbf{x},y)\in S} h_\theta^2(x) c_{\mathbf{x}} + \lambda_\theta}. \quad (6)$$

Finally, the third learning method is based on Markov Chain Monte Carlo (MCMC) [24]. MCMC is an implementation of Bayesian inference technique that generates the distribution of parameters. The conditional posterior distributions for each parameter is generated from $\mathcal{N}(\widetilde{\mu}_\theta, \widetilde{\sigma}_\theta^2)$:

$$\widetilde{\mu}_\theta = \frac{\alpha\theta \sum_{i=1}^{n} h_\theta^2(\mathbf{x}_i) c_{\mathbf{x}} + \alpha \sum_{i=1}^{n} h_\theta(\mathbf{x}_i) e_i c_{\mathbf{x}} + \mu_\theta \lambda_\theta}{\alpha \sum_{i=1}^{n} h_\theta^2(\mathbf{x}_i) c_{\mathbf{x}} + \lambda_\theta}, \quad (7)$$

$$\widetilde{\sigma}_\theta^2 = \frac{1}{\alpha \sum_{i=1}^{n} h_\theta(\mathbf{x}_i)^2 c_{\mathbf{x}} + \lambda_\theta}. \quad (8)$$

where $e_i = y_i - \hat{y}_i(\mathbf{x}_i|\theta)$ is the error term of ith record, α is the precision of the likelihood, μ_θ is the prior mean of the parameter, and λ_θ is the prior precision of the parameter. Please note that the solution would be equivalent to ALS when $\theta^* = \widetilde{\mu}_\theta$ with $\alpha = 1$ and $\mu = 0$.

Among these learning methods, both the ALS method and the MCMC method update the model parameters according to entire errors. Moreover, the MCMC method is an advanced version of the ALS method by sampling the model parameters. Therefore, although it is possible to experiment with all of them, we opt for reporting the result of the MCMC method in this paper. Another advantage of the MCMC method is that it can learn recommendation model without giving the external parameters such as learning rate and the regularization term.

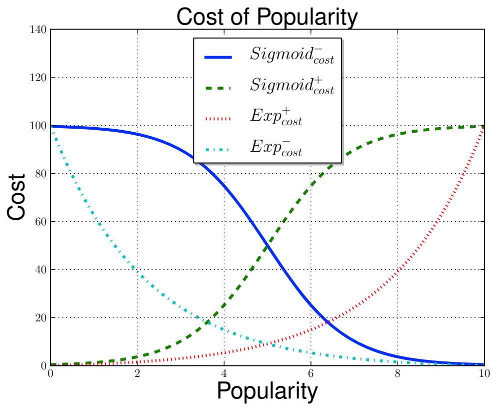


Fig. 1. Four possible popularity weighting functions.

3) Popularity cost: By using the cost-sensitive FM, we are able to enhance the effects of unpopular music. We experiment with the following two weighting functions for adjusting the weights of each music. Both of them are modified by two well-known functions.

The first one is the *sigmoid cost*. We convert the primitive sigmoid function into a flexible form and re-scale the curve from 0 to the given maximum popularity (e.g. $p_{\max}$) of data. Let p_x is denoted as the popularity of an instance $\mathbf{x}$:

$$Sigmoid_{cost}(p_{\mathbf{x}}) = \frac{c_{\max}}{1 + e^{\pm(4e\frac{p_{\mathbf{x}}}{p_{\max}} - 2e)}}, \tag{9}$$

where $c_{\max}$ is the maximum value of items' popularity, and you can observe that $\frac{p_{\mathbf{x}}}{p_{\max}}$ is the percentage. We use $Sigmoid^{+}_{cost}$ to denote the function that is positive favor, and $Sigmoid^{-}_{cost}$ to denote the one that is negative favor, respectively.

The second one is the *exponential cost*. In a similar way, another weighting function is extended by the exponential function plus the given popularity $p_{\mathbf{x}}$:

$$Exp_{cost}(p_{\mathbf{x}}) = e^{\pm(\frac{p_{\mathbf{x}}}{p_{\max}})\log w_{\max}}. \tag{10}$$

We use Exp^{+}_{cost} and Exp^{-}_{cost} to denote the positive and negative favor functions, respectively.

For the two proposed popularity weighting functions, we assume that $w_{\max} = 100$ and $p_{\max} = 10$. We plot the corresponding cost of corresponding popularity for each level as the shown in Figure II-B3. Take $Sigmoid^{+}_{cost}$ as the example, the items that are located on the distribution of high popularity will donate greater effects upon the loss function, while the $Sigmoid^{-}_{cost}$ takes care of the unpopular items.

There still exists various functions could be utilized as the weighting function, such as the Gompertz function. In this paper, we report only the two kinds of weighting functions. In fact, the proposed cost-sensitive FM is constructed on the instance-level. Therefore we are able to build any shape of function according to different user needs.

III. Experiment

We aim at mitigating the problem of long tail without compromising the accuracy of the recommendation. To realize this goal, we manipulate the loss function of the FM model to make it cost-sensitive. Different kinds of loss function are examined in the experiments, including sigmoid and exponential. The performance of the original loss function (i.e. without cost-sensitive) is also considered to serve as a baseline method.

A. Data Description

We collect a real-world dataset from KKBOX, which is a well-known digital music service company in Asia. The dataset contains 1,800,000 listening records that cover 5,000 users and 30,000 songs. For experimental settings, we first split the dataset into training set and testing set following 80/20 rules. That is, we keep full listening history for 80% users. For the remain 20% users, we randomly split their listening records into two parts. Given half of listening records for a user, we seek to predict which songs the user would like to listen to (i.e. the missing half).

B. Performance Measure

We apply two metrics to evaluate recommendation performance: Popularity and Recall. Popularity is used to measure how popular a song is. Here we define the popularity as the amount of listening users who have listened to that song. Recall is defined as one minus the proportion of missing records returned by the recommendation system.

It is important to consider the recall measure for a recommendation problem. In the real world scenario, recommending unpopular songs usually hurts the performance, but we argue that the goal of recommendations is to help users to find out the songs they never listen to before but they actually like. Hence, recall is a good measurement to examine whether the proposed recommendation algorithm is able to generate a recommendation list that covers the user likes, and remains stable performance as well.

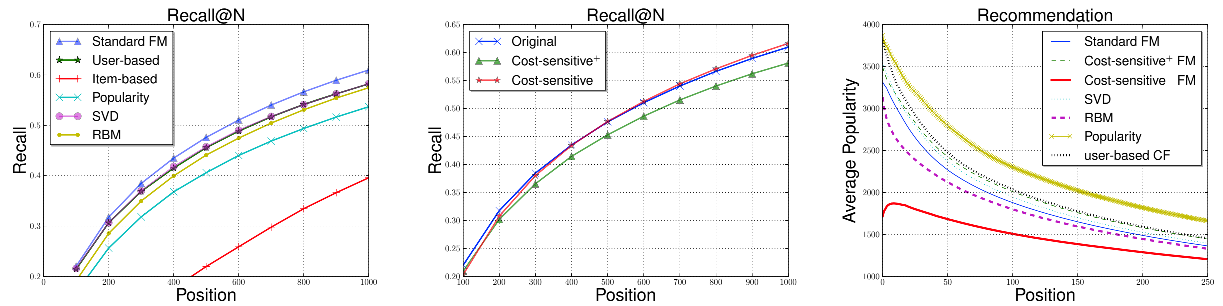
C. Evaluation Result on the Recall Measure

Our experiments are conducted on pure user-item ratings. To validate the effectiveness of the FM model, we first compare the performance of the standard FM with a number of widely-used recommendation algorithms via the recall measurement, and check the performance of proposed cost-sensitive Factorizaiton Machine later on.

We briefly illustrate the compared algorithms below. They are derived from diverse kinds of frameworks including Collaborative Filtering (CF), Matrix Factorization and Neural Network.

Popularity-based Recommendation is a baseline algorithm that always generates the most popular music as the recommendations.

User-based/item-based CF is one of the most popular algorithms for a variety of recommendation problems. This kind of algorithm predicts the rating of specific song based on the neighbors' ratings. In general, there are two main categories of the CF method: User-based CF and Item-based CF. We implemented these two algorithms by using the extended version of cosine similarity computation [25].



(a) Recall Measure (b) Recall Measure (c) Popularity Measure

Fig. 2. The performance of different recommendation algorithms in terms of a number of performance measures.

Singular Value Decomposition (SVD) is a traditional matrix factorization technique that factors any $m \times n$ matrix M into three matrices $M\Sigma V^T$ [26]. We also evaluate the performance of SVD++, an extended version of SVD-based latent factor models by integrating implicit feedbacks [27].

Restricted Boltzmann Machines (RBM) is a generative stochastic neural network that can learn a probability distribution over given data and can be successfully applied to the recommendation problem [28], [29].

The experimental results are shown in the Figure 2a. The bad performance of item-based CF is due to the long-tails and data sparsity problem. Except for item-based CF, most methods reach comparable result with others. Among the considered methods, FM obtains the highest performance, which shows that FM can be a competitive framework for this task.

Regarding to the cost-sensitive FM, we report the performance both on the positive and negative correlation of popularity weighting functions. As shown in Figure 2b, the cost-sensitive FM is able to reach almost the same recall rate as standard FM.

D. Evaluation Result on the Popularity Measure

As shown in Figure 2c, the x-axis represents the recommendations at the cut-off position N and the y-axis is the corresponding average popularity. The experimental result shows that, we can successfully lower the popularity of the recommended music, especially for the top recommendations. Certainly, the popularity-based approach contains the highest average cover users in every cut-off positions. The other approaches still suffer from the problem of always recommending the popular music. Considering our two proposed approaches, the positive-favor cost-sensitive FM produces the relative higher popular music list than the standard FM, while the negative-favor cost-sensitive FM remarkably reduce the average cover users from about 3,300 to 1,800 in top recommendations. Note that the cost sensitive method is effective in terms of both the recall measure and the popularity measure.

IV. Conclusions

We propose a new recommendation algorithm called cost-sensitive factorization machine, which is a general framework that can incorporate any kinds of predefined cost criterion. This algorithm can help address the phenomenon of long tail for any recommendation problems by utilizing the proposed popularity weighting functions. The high weighting of unpopular items makes the cost-sensitive FM model feasible to pay attention on the long tailed items, and further reduce the popularity bias of recommendation without compromising the recall rate. We experiment with these weighting functions to weight different instances differently and show by experiment the effectiveness of the proposed method.

Acknowledgment

This research is supported by a grant from KKBOX and a grant from the Ministry of Science and Technology under contract 102-2420-H-004-052-MY2.

References

[1] J. L. Herlocker, J. A. Konstan *et al.*, "Evaluating collaborative filtering recommender systems," *ACM Trans. Information System*, vol. 22, no. 1, pp. 5–53, 2004.

[2] H. Yin, B. Cui *et al.*, "Challenging the long tail recommendation," *CoRR*, 2012.

[3] Y.-J. Park and A. Tuzhilin, "The long tail of recommender systems and how to leverage it," in *Proc. RecSys*, 2008.

[4] J. Oh, S. Park *et al.*, "Novel recommendation based on personal popularity tendency." in *Proc. ICDM*, 2011.

[5] H. Steck, "Item popularity and recommendation accuracy," in *Proc. RecSys*, 2011.

[6] L. Shi, "Trading-off among accuracy, similarity, diversity, and long-tail: A graph-based recommendation approach," in *Proc. RecSys*, 2013.

[7] M. Zhang, N. Hurley *et al.*, "A double-ranking strategy for long-tail product recommendation," *WI-IAT*, 2012.

[8] K. Lee and K. Lee, "My head is your tail: Applying link analysis on long-tailed music listening behavior for music recommendation," in *Proc. RecSys*, 2011.

[9] S. Rendle, "Factorization machines with libFM," *ACM Trans. Intelligent Systems and Technology*, vol. 3, no. 3, 2012.

[10] ——, "Factorization machines," in *Proc. ICDM*, 2010.

[11] S. Rendle, Z. Gantner *et al.*, "Fast context-aware recommendations with factorization machines," in *Proc. SIGIR*, 2011.

[12] C.-M. Chen, M.-F. Tsai, J.-Y. Liu, and Y.-H. Yang, "Music recommendation based on multiple contextual similarity information," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intelligence*, 2013.

[13] ——, "Using emotional context from article for contextual music recommendation," in *Proc. ACM MM*, 2013.

[14] C. Elkan, "The foundations of cost-sensitive learning," in *Proc. IJCAI*, 2001.

[15] G. M. Weiss, K. McCarthy, and B. Zabar, “Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs?” in *Proc. DMIN*, 2007.

[16] X.-Y. Liu and Z.-H. Zhou, “The influence of class imbalance on cost-sensitive learning: An empirical study.” in *Proc. ICDM*, 2006.

[17] Y. Sun, M. S. Kamel *et al.*, “Cost-sensitive boosting for classification of imbalanced data,” *Pattern Recogn.*, 2007.

[18] P.-L. Chen and H.-T. Lin, “Active learning for multiclass cost-sensitive classification using probabilistic models,” in *Proc. TAAI*, 2013.

[19] C.-L. Li and H.-T. Lin, “Condensed filter tree for cost-sensitive multi-label classification,” in *Proc. ICML*, 2014.

[20] ——, “One-sided support vector regression for multiclass cost-sensitive classification,” in *Proc. ICML.*

[21] N. Srebro and T. Jaakkola, “Weighted low-rank approximations,” in *Proc. ICML*, 2003.

[22] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in *Proc. ICDM*, 2008.

[23] Y. Shi *et al.*, “Nontrivial landmark recommendation using geotagged photos,” *ACM Trans. Intelligent Systems and Technology*, vol. 4, no. 3, 2013.

[24] S. R. Christoph Freudenthaler, Lars Schmidt-Thieme, “Bayesian factorization machines,” in *Workshop on Sparse Representation and Low-rank Approximation, Neural Information Processing Systems*, 2011.

[25] C.-M. Chen, M.-F. Tsai *et al.*, “Using emotional context from article for contextual music recommendation,” in *Proc. ACM MM*, 2013.

[26] J. E. R. V. Hernandez and A. Tomas, “Restarted lanczos bidiagonalization for the svd in slepc,” STR-8, Tech. Rep., 2007.

[27] Y. Koren, “Factorization meets the neighborhood: A multifaceted collaborative filtering model,” in *Proc. SIGKDD*, 2008.

[28] G. E. Hinton, “A practical guide to training restricted boltzmann machines.” in *Neural Networks: Tricks of the Trade (2nd ed.)*, 2012.

[29] R. Salakhutdinov, A. Mnih, and G. Hinton, “Restricted boltzmann machines for collaborative filtering,” in *Proc. ICML*, 2007.