Click-Through Prediction for Sponsored Search Advertising with Hybrid Models

Xingxing Wang [1,5,*] Shijie Lin [1,5,*] Dongying Kong [3,5,*] Liheng Xu [2,5,*]
Qiang Yan [2,5,*] Siwei Lai [2,5] Liang Wu [1,5,7] Alvin Chin [7]
Guibo Zhu [2,5] Heng Gao [5,6] Yang Wu [1,5] Danny Bickson [4]
Yuanfeng Du [1,5] Neng Gong [1,5] Chengchun Shu [3,5] Shuang Wang [3,5]
Kang Liu [2] Shuren Li [1] Jun Zhao [2] Fei Tan [1,5,†] Yuanchun Zhou [1,‡]

[1]Computer Network Information Center, Chinese Academy of Sciences, Beijing
[2]National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
[3]The Institute of Computing Technology, Chinese Academy of Sciences, Beijing
[4]Carnegie Mellon University, 5000 Forbes Ave Pittsburgh, PA
[5]Graduate University of Chinese Academy of Sciences
[6]Institute of Automation, Chinese Academy of Sciences, Beijing
[7]Nokia Research Center, Beijing

ABSTRACT

In this paper, we report our approach of KDD Cup 2012 track 2 to predicting the click-through rate (CTR) of advertisements. To accurately predict the CTR of an ad is important for commercial search engine companies for deciding the click prices and the order of impressions. We first implemented three existing methods including Online Bayesian Probit Regression (BPR), Support Vector Machine (SVM) and Latent Factor Model (LFM). In order to fully exploit the training set, several Maximum Likelihood Estimation(MLE)-based methods are then proposed to model the instances which appear frequently in the training set. Each of the individual models is optimized by selecting the most descriptive features. We propose a rank-based ensemble method which greatly improves the results of our model and our final submission is based on BPR, SVM and MLE.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Application

*These authors contributed equally to this work
†Team leader
‡zyc@cnic.cn

Keywords

KDD Cup 2012, Online Advertising, Sponsored Search, Feature Engineering, Ensemble Method

1. INTRODUCTION

KDD Cup is a famous data mining competition in both industry and academia. There are two tracks in KDD Cup 2012, of which the first task is about the recommendations and predictions in social networks and social media, and the second task is about predicting the click-through rate of online advertising. Tencent Corporation released a large real dataset in the KDD Cup competition, which was obtained from the user logs of Tencent's commercial search engine called SOSO [10]. For the second track, the challenge is not only about the processing for such big data, but is also about developing the most accurate predictor for the search engine.

In this task, participants are expected to accurately predict the click-through rate (pCTR) of ads, and the search session logs, which capture the interaction between a user and the search engine, are provided. There are three types of data sets, which include training data, additional data and testing data. Training and testing data both contain the following items: the user, the query issued by the user,and some ads returned by the search engine. The training set also contains the impressions and clicks of each ad. Moreover, each ad and each user have some additional properties, which are included in the additional data set. The big data challenge in this task is, given 149,639,105 instances as the training data, we are required to predict the click-through rate for a test set with 20,297,594 instances.

The challenge can be viewed as a regression task because we can model the CTR prediction task using the information like the titles, the descriptions and the user profiles as features. On another hand, we can use the number of an ad's impressions and the number of the ad's clicks to calculate

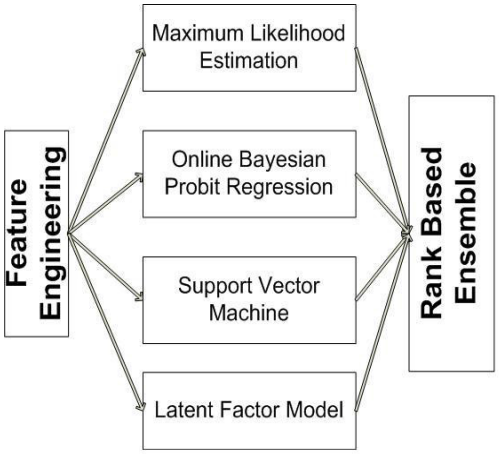

Figure 1: The architecture of our proposed models.

the click-through rate based on the prediction results of the test data, i.e., for every impression, the user may click or not click the ad, which can be represented as 0 (a user does not click the ad) and 1 (the user clicks the ad). Thus, the problem can also be seen as a binary classification [11]. In addition, several recent works have exploited collaborative filtering-based methods to rank the ads[1, 12], which is suitable in the competition. We have tried several approaches to predict the CTR including the regression models, the classification models and the latent factor models, of which we have selected some useful ones to introduce in Section 3.

The metric for the performance of the prediction is the Area under an ROC Curve (AUC), which was proposed in [3]. The metric is concerned only about the CTR order of the testing data, so we rank them instead of the real values of their CTR. The AUC value will always be between 0 and 1.0, and the random guessing for the ranking of the results will produce AUC of 0.5, therefore our prediction results must have an AUC score higher than 0.5.

At the Graduate University of Chinese Academy of Sciences, three teams from different labs are engaged in this competition. Initially, different teams developed their models individually. In order to avoid overlap between the teams, each team needed to use some new features, which were different from other teams. At the end of this competition, we used the rank-based technique to blend the results from different models. The ensemble result provided us with the 3rd place on the private leaderboard in this competition.

Figure 1 shows the framework of our models. We first extracted and produced many descriptive features based on a comprehensive feature engineering method. Then four models were applied to the training set by using the extracted features. Finally, we proposed a rank-based ensemble method, which normalized the results of the four models and generated the final results.

The paper is structured as follows. In Section 2, we give an overview about all the features we used in this competition. Section 3 focuses on the individual models proposed by the different teams, which include Bayesian probit regression, latent factor model, support vector machine and maximum likelihood estimation. In Section 4, the ensemble method is presented. In Section 5, we discuss the contribution of each model for the final performance, so that we can have a good understanding of the importance of different features in the training data as well as characteristics of each model. The last section concludes the paper.

2. FEATURES

We propose to use two features for the models. The first feature is directly extracted from the training data, which we call the original features, while the second feature is the synthetic features.

- *Original Features*: The original feature set contains discrete features and continuous features. The discrete features are the unique ID of each ad, advertiser, query, keyword, title, description, token, user and the gender and age of a user, the position of an ad and the displayed url. The continuous features are the click-through rates of each value of the discrete features, i.e., when a discrete feature is activated, the corresponding click-through rate is also activated and adopted as a continuous feature.
- *Synthetic Features*: First, we join any two original discrete features with each other and use them as synthetic features. We also test some 3-tuple features but only the tuple of QueryID_AdID_UserID is used, since most 3-tuple features are too sparse and seldom activated. Second, we join the original discrete features with each of the tokens. We further add the position information to the original discrete features and 2-tuple features to generate the position-based features. In addition, the bigram features are also adopted for analyzing the queries, titles and descriptions.

3. INDIVIDUAL METHODS

In this section, we present the individual models we use and their leaderboard results respectively. Some other models, like General Linear Regression, are omitted since they contribute little to the final results.

3.1 Online Bayesian Probit Regression

The Online Bayesian Probit Regression was introduced by the researchers of Microsoft [7]. It was used to predict the CTR of sponsored search advertising in the Bing search engine and performed very well. The model is based on a probit regression model that maps discrete or real-valued input features to probabilities. It maintains Gaussian beliefs over the weight of the model and performs Gaussian updates derived from approximate message passing.

3.1.1 LDA Preprocess

The tokens in the training data are words and a string of tokens can be treated as a document. Therefore, we use the LDA (Latent Dirichlet Allocation) model [2], which was introduced for the purpose of document modeling, to preprocess the token files. We assume the following generative process for a string of tokens d:

1. Choose $\theta_d \sim Dir(\alpha)$,a Dirichlet distribution with parameter a.
2. for each of the N tokens in d:
   (a) Choose a topic $z_n \sim Multinomial(\theta_d)$, a mulitinomial distribution of parameter θ_d.
   (b) Choose a token t_n from $p(t_n|z_n,\beta)$, a multinomial probability conditioned on the topic z_n.

For corpus D of all strings of tokens, we can obtain the probability:

$$p(D|\boldsymbol{\alpha},\boldsymbol{\beta}) = \prod_{d=1}^{M} \int p(\theta_d|\alpha) \cdot \left(\prod_{n=1} N_d \sum_{z_{d,n}} p(z_{d,n}|\theta_d)\, p(t_{d,n}|z_{d,n},\beta) \right) d\theta_d \quad (1)$$

We use the Gibbs sampling method to infer and obtain a parameter vector θ_d for a string of tokens d. We use the vector θ_d as the features of the string of tokens d.

3.1.2 Features

We apply the LDA method described above to preprocess the query tokens,the purchased keyword tokens and the title tokens. In addition, the features like the IDs, the user information and the similarities between the query and ad are also included. We use three kinds of similarities: the similarity between query and purchased keyword (sim_{q_k}), the similarity between query and ad title (sim_{q_t}), and the similarity between query and ad description (sim_{q_d}). We represent them by three 2-dimensional vectors:
$(sim_{q\_k}, 1 - sim_{q_k})$
$(sim_{q\_t}, 1 - sim_{q\_t})$
$((sim_{q\_t}, 1 - sim_{q\_t})$
The final feature vector x is composed by many groups of vectors: $\boldsymbol{x} = \left(\boldsymbol{x}_1^{\mathrm{T}}, \boldsymbol{x}_2^{\mathrm{T}}, \ldots, \boldsymbol{x}_N^{\mathrm{T}}\right)^{\mathrm{T}}$ where

$$\boldsymbol{x_i} = \begin{pmatrix} x_{i,1} \\ x_{i,2} \\ \vdots \\ x_{i,M_i} \end{pmatrix}, \sum_{j=1}^{M_i} x_{i,j} = 1 \quad (2)$$

3.1.3 Probability Model and Factor Graph

The Bayesian Probit Regression model is based on a generalized model linear mode with a probit link function:

$$p(y|\boldsymbol{x},\boldsymbol{w}) = \Phi\left(\frac{y \cdot \boldsymbol{w}^{\mathrm{T}} \cdot \boldsymbol{x}}{\beta}\right) \quad (3)$$

Here $\Phi(t) = \int_{-\infty}^{t} \mathcal{N}(s;0,1)\, ds$ is the standardized cumulative Gaussian density (probit function) which serves as the inverse link function mapping the output of the linear model in $(-\infty,\infty)$ to $(0,1)$, where $\boldsymbol{x}$ denotes the feature vector and $\boldsymbol{w}$ denotes weight vector. As the feature vector $\boldsymbol{x}$ is composed of many groups of vectors, the weight vector is grouped according to the feature vector: $\boldsymbol{w} = \left(\boldsymbol{w}_1^{\mathrm{T}}, \boldsymbol{w}_2^{\mathrm{T}}, \ldots, \boldsymbol{w}_N^{\mathrm{T}}\right)^{\mathrm{T}}$. $y \in \{-1,+1\}$, where -1 represents a non-click, and $+1$ represents a click. The parameter β scales the steepness of the inverse link function.

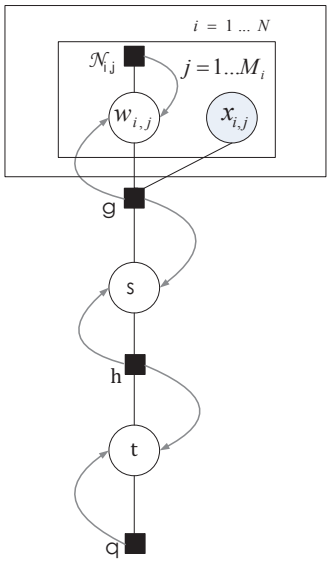

Figure 2: Factor graph model of Bayesian probit regression with message flow.

In order to arrive at a Bayesian online learning algorithm, based on the above model, the factorizing Gaussian prior distribution over the weights of the model is postulated and two latent variables s,t are introduced:

$$p(\boldsymbol{w}) = \prod_{i=1}^{N} \prod_{1}^{M_i} \mathcal{N}(w_{i,j}; \mu_{i,j}, \sigma_{i,j}^2) \quad (4)$$

$$s = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} \qquad p(s) = \mathcal{N}\left(s|\boldsymbol{x}^{\mathrm{T}}\boldsymbol{\mu}, \left(\boldsymbol{x}^2\right)^{\mathrm{T}}\boldsymbol{\sigma}^2\right) \quad (5)$$

$$p(t|s) = \mathcal{N}\left(t; s, \beta^2\right) \quad (6)$$

$$p(y|t) = \Phi(y \cdot t) \quad (7)$$

therefore,the joint density function $p(y,t,s,\boldsymbol{w}|\boldsymbol{x})$ can be factorized as

$$p(y|t)\, p(t|s)\, p(s|\boldsymbol{x},\boldsymbol{w})\, p(\boldsymbol{w}) \quad (8)$$

This distribution can be understood in terms of the following generative process, which is also reflected in the factor graph in Figure 2:

- Factor $\mathcal{N}_{i,j}$: Sample weight $w_{i,j}$ from the Gaussian prior $p(w_{i,j}) = \mathcal{N}_{i,j}\left(\mu_{i,j}, \sigma_{i,j}^2\right)$.
- Factor g: Calculate the score s for $\boldsymbol{x}$ as the inner product $\boldsymbol{w}^{\mathrm{T}}$ such that $s = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$.
- Factor h: Add zero-mean Gaussian noise to obtain t from s,such that $p(t|s) = \mathcal{N}\left(t; s, \beta^2\right)$.
- Factor q: Determine y by a threshold on the noisy score t at zero, such that $p(y|t) = \Phi(y \cdot t)$.

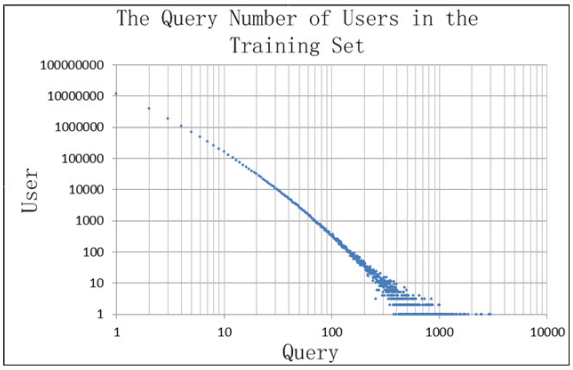

Figure 3: User Distribution by Query in the Training Set.

To calculate the posterior over weight $\boldsymbol{w}$ in factor graph Figure 2, expectation propagation, an approximation message passing algorithm, is used. After derivation by using expectation propagation,the posterior weight $\widetilde{\boldsymbol{w}} = \left(\widetilde{\boldsymbol{\mu}}, \widetilde{\boldsymbol{\sigma}}^2\right)$ can be calculated as:

$$\widetilde{\mu}_{i,j} = \mu_{i,j} + yx_{i,j} \cdot \frac{\sigma_{i,j}^2}{\Sigma} \cdot \nu\left(\frac{y \cdot \boldsymbol{x}^{\mathrm{T}}\boldsymbol{\mu}}{\Sigma}\right) \tag{9}$$

$$\widetilde{\sigma}_{i,j}^2 = \sigma_{i,j}^2 \cdot \left[1 - x_{i,j} \cdot \frac{\sigma_{i,j}^2}{\Sigma^2} \cdot \omega\left(\frac{y \cdot \boldsymbol{x}^{\mathrm{T}}\boldsymbol{\mu}}{\Sigma}\right)\right] \tag{10}$$

where:

$$\Sigma^2 = \beta^2 + \left(\boldsymbol{x}^2\right)^{\mathrm{T}} \boldsymbol{\sigma}^2 \tag{11}$$

$$\nu(t) = \frac{\mathcal{N}(t;0,1)}{\Phi(t;0,1)}, \omega(t) = \nu(t) \cdot [\nu(t) + t] \tag{12}$$

These equations lead to anatural online learning algorithm in which the parameter of the prior weight distribution $\left(\boldsymbol{\mu}.\boldsymbol{\sigma}^{\boldsymbol{2}}\right)$ are initialized to reflect any prior information. For the first training example$(\boldsymbol{x}, y)$, the posterior parameter$\left(\widetilde{\boldsymbol{\mu}}, \widetilde{\boldsymbol{\sigma}}^2\right)$ are calculated. After that, the previously obtained posterior is used as the prior for the next update, i.e., $\boldsymbol{\mu} \leftarrow \widetilde{\boldsymbol{\mu}}$ and $\boldsymbol{\sigma}^2 \leftarrow \widetilde{\boldsymbol{\sigma}}^2$

3.1.4 Result

Using the features described in Section2, and applying them to train the online Bayesian probit regression model. An AUC of 0.7902 from the test data is achieved.

3.2 Support Vector Machine

As discussed in the introduction, the CTR prediction task can also be formulated as a binary classification problem, i.e., predicting whether a user will click the ad given the features discussed in Section 2. In order to predict the click-through rate of a specific advertisement, we can calculate the click percentage of total impressions by predicting whether a user will click the ad. We use SVMperf [8], an implementation that can optimize a number of multivariate performance measures to train a binary classifier which incorporates the AUC as optimization criterion. We select a subset of the features discussed in Section 2 for the classification model including discrete features and continuous features.

Table 1: Data Analysis of Training and Testing Data

	in training set	not in training set
	User ID	csr = 57.72%
in testing	1379733	1883948
not in testing	20643814	0
	Query ID	csr = 55.79%
in testing	1680669	2121309
not in testing	22441407	0
	Ad ID	csr = 9.62%
in testing	271159	28853
not in testing	370548	0
	Advertiser ID	csr = 3.62%
in testing	10753	404
not in testing	4094	0
	Keyword ID	csr = 12.45%
in testing	433744	61677
not in testing	754345	0
	Display URL	csr = 4.23%
in testing	16943	749
not in testing	9330	0
	Description ID	csr = 24.19%
in testing	744419	237566
not in testing	2189682	0
	Title ID	csr = 27.99%
in testing	947049	315449
not in testing	2788747	0

When training the model, we find that it takes a long time to converge as the data size is quite large. So we propose to use a filter to accelerate the training process. First we perform a comprehensive data analysis on the datasets. Table 1 depicts some of the results and the corresponding cold start ratios for each feature. In Table 1, csr denotes the cold start rate of each feature, and the entries represent the number of the data which occurs or does not occur in the training/testing set. Figure 3 depicts the query number distribution against users. We can see that the distribution is heavy tailed, and so are the distributions of other features like impressions and the term frequency of queries. Based on the analysis in Table 1 and Figure 3, we use a simple filter for feature selection. It filters out features that do not exist in the testing set and the features with low frequency (that occurred less than 20 times).

Our experiments show that these two strategies can effectively speed up the training process of the classification model without loss of performance.

In addition, we use a linear kernel and randomly separate the training data into 10 parts to train 10 models independently, due to the huge size of training set and large requirement of memory usage by SVMperf. Finally we use the average score from the predictions of 10 models as our final result.

When using SVMperf, we find that using UserIDs as binary features can cause severe over fittings. Therefore we remove user features from our original model and simply multiply the average click rate of each user to the prediction score of

our final model. If a user never clicks any advertisement, we multiply a score of 0.015 for smoothing. This simple strategy is quite robust and can approximately improve our leaderboard score by 0.007.

3.3 Latent Factor Model

Latent Factor Model [9] and its variances are probably the most popular collaborative technique that demonstrates a significant performance in rating prediction of the recommendation task. In Latent Factor Model, both users and items are projected to a latent factor space of dimension K; and the prediction is computed by the dot product of the user latent feature vector p_u and the item latent feature vector q_i,

$$r_{ui}^{\sim} = p_u q_i \tag{13}$$

where p_u and q_i can be learned by minimizing the following loss function via stochastic gradient descent:

$$min \sum (r_{ui} - r_{ui}^{\sim}) + \lambda(||p_u||^2 + ||q_i||^2) \tag{14}$$

CTR prediction can also be converted to a rating prediction problem, if a user clicks an ad, we can map the user and the ad into a latent factor space. We have more information about the ad and user, such as the user's demographic and the ad's title, description, advertiser and query. Thus, we use a Feature-based Latent Factor Model, just like SVDFeature [4], where we project the user and ad features into latent factor space and all the features we use are the discrete user and ad features described above.

For the tokens, we select top N tokens for each ad via TF-IDF. For the many new users appear in the test dataset, we map the user's demographical information like the age and gender into the latent factor space to describe this kind of users. Like the Latent Factor Model, we add these bias for each feature. The exact model is as follows:

$$r_{ui}^{\sim} = \mu + \sum_{f_i \in F_i} b_{f_i} + \sum_{f_u \in F_u} b_{f_u} + (\sum_{f_i \in F_i} q_{f_i})^T (\sum_{f_u \in F_u} p_{f_u}) \tag{15}$$

F_i indicates the set of all ad features while F_u is the set of all user features. Similarly, we can learn the parameters by using stochastic gradient descent optimization to minimize the quadratic loss function (See equation 16), and the complexity of this model is linear with the number of impressions.

$$\begin{aligned} \mathrm{I} = min \sum (r_{ui} - r_{ui}^{\sim})^2 + \lambda(\sum_{f_i \in F_i} ||b_{f_i}||^2 + \sum_{f_u \in F_u} ||b_{f_u}||^2 \\ + \sum_{f_i \in F_i} ||q_{f_i}||^2 + \sum_{f_u \in F_u} ||p_{f_u}||^2) \end{aligned} \tag{16}$$

In our experiment, we can get 0.771 on the leader board; all the tunable parameters we use are listed below:

- Learning rate decay (0.9)
- Learning rate (5e-3)
- Weight factor (1e-3)
- Iterations (50)

3.4 Feature-Based Maximum Likelihood Estimation

In order to fully exploit the big training set, we propose to use the Maximum Likelihood Estimation (MLE) to model the distributions of different features, i.e., we compute the click-through rates for each of the features. For MLE, we use the original discrete features, 2-tuple features and position-based features.

Since the click-through rate (CTR) of ads decreases significantly according to its positions, we used a position-normalized statistic to account for this position bias known as clicks over expected clicks (COEC)[5]:

$$COEC = \frac{\sum_{r=1}^{R} C_r}{\sum_{r=1}^{R} i_r \times CTR_r} \tag{17}$$

The numerator is the total number of clicks received by a feature variable and the denominator can be interpreted as the expected clicks that an average ad would receive after being impressed i_r times at rank r, and CTR_r is the average CTR for each position in the result page(up to R).

When estimating the click-through rate of a new instance, we first retrieve the CTR for each feature of it, and assemble them to produce the estimated results. In addition, we propose some penalty rules to revise the results. The ensemble method and the penalty rules are discussed below.

3.4.1 Weighted Ensemble of Individual Features

Given all the features, our goal is to get the click-through rate of each feature. We use f_i to denote feature i, and use $fctr_i$ to denote the click-through rate of f_i. Based on the basic intuition of maximum likelihood estimation, it is estimated by computing the percentage of the positive instances in all the instances that contain f_i. By individually using the features, we can get an AUC value for each feature on the validation set. The AUC value w_i is then used as the f_i's weight.

In order to leverage the position information, we separately calculate the click-through rate $fctr$ according to their positions as follows:

$$fctr_{i,j} = fctr_i \times \frac{CTR_j}{\sum_{r=1}^{|R|} CTR_r} \tag{18}$$

where $fctr_{i,j}$ is the feature click-through rate of the feature i at the position of j and $fctr_i$ is the click-through rate of the feature i. CTR_j denotes the click-through rate of the ads which has the position of j and R is the set of positions.

When predicting the click-through rate of a test instance, we first produce the feature set of it. Assuming that F is the collection of the instance's features, the click-through rate is estimated as follows:

$$CTR = \sum_{i=1}^{|F|} (fctr_i \times w_i) \tag{19}$$

3.4.2 Penalty Rules

Age \ Gender	Unknown	Male	Female	Total
(0, 12]	0.041684	0.041937	0.041666	0.041802
(12, 18]	0.044643	0.043964	0.041407	0.042994
(18, 24]	0.042368	0.039424	0.042508	0.040824
(24, 30]	0.041116	0.037291	0.042872	0.039807
(30, 40]	0.041385	0.042602	0.047287	0.044667
(40, inf)	0.044224	0.047588	0.052256	0.049530
Total	0.041970	0.040973	0.043735	0.042173

Figure 4: The average CTR of different users.

In order to improve the prediction ability of MLE, we analyze the training set. Figure 4 depicts the average click-through rate of different users. The color means the user density of each entry, e.g., the users who are male and are between 24 and 30 years old occupy the largest share among all users. By further analyzing the unknown users, we find that the users with their ID equal to 0 have an average CTR of 0.024926, while that of the others is 0.038308. Based on the above observations, we find that when some specific pattern occurs, the click-through rate drops dramatically. Thus, we obtain some penalty rules to leverage the observations.

We name these patterns as penalty patterns and there are two kinds of penalty patterns which are described below.

- Null User Pattern: By analyzing the training dataset, we find the CTR of the instances which contains a null user, i.e., the users with an ID as zero, are much lower than the others.
- User Ad Query (UAQ) Pattern: By analyzing the training dataset, we find that for a specific kind of 3-tuple feature $User\_Ad\_Query$, when the CTR of which is less than the threshold, the instances which contain them have a lower click-through rate.

For those instances that contain any penalty rules, the click-through rates are post-processed as follows, where CTR denotes the original CTR and CTR' denotes the new CTR. $\#IMP$ represents the number of impressions of the corresponding patterns which activates the postprocess.

$$CTR' = \frac{CTR}{log(\#IMP) + 1} \tag{20}$$

4. RANK-BASED BLENDING

In order to leverage the results of the individual models discussed in Section 3, an ensemble method is introduced to combine them, which is similar to the rank fusion method proposed in [6] and the voting method Borda Count. It is easy to combine the results by linearly ensembling all the results. The ensemble results, however, are not good enough. Because of the following two reasons which may bring in noise. The first reason is that for a given model, the differences of the predicted click-through rate (pCTR) between different instances are small, which means an outlier may affect the final ranking if the results are linearly combined. The second observation is that some models have a higher sum of overall pCTR than other models. This may result in some models that are much more influential than others and even dominate the final results. In order to solve the problems, we propose a pCTR blending method for tackling the outlier problem and the imbalance between different individual models. The blending method is based on the ranking of pCTR.

Table 2: The results and improvement ratio on the public leaderboard of the blending model and the individual models.

	Result	Ratio
BPR	0.7902	1.6705%
SVM	0.7865	2.1488%
LFM	0.7710	4.2023%
MLE	0.7924	1.3882%
BLENDING	0.8034	0.0%

Assuming that $rank$ is the rank number of an instance by ranking all of them in a descending order, thus, the instances with a higher CTR prediction results will get a larger $rank$. By using the ranking scheme, the differences between the prediction results and the contributions of different models are normalized, thus easing the influence of outliers. The ranking scheme has a 0.00493 improvement on the leader board.

Finally, we propose to use the harmonic mean to ensemble the results generated by individual models as follows, which generates the final blending submission:

$$\begin{aligned} rank_i &= \frac{m}{\frac{1}{rank_1(i)} + \frac{1}{rank_2(i)} + \ldots + \frac{1}{rank_m(i)}} \\ &= \frac{m}{\sum_{j=1}^{m} \frac{1}{rank_j(i)}} \end{aligned} \tag{21}$$

By using the harmonic mean, our result on the leaderboard increases to 0.00042 and the ensemble method improves by about 0.013 on average over the individual models on the leaderboard. Table 2 illustrates the results of the individual models on the public leaderboard and the column of Ratio denotes the improvement ratio of the blending model.

5. CONCLUSION

In this paper, we discuss our approach to KDD Cup 2012, track 2. As the data size is quite large, our basic intuition is to try different methods to better model the user click behaviors. We proposed to use four individual models and each of them performs well on the prediction task. In the final ensemble for private leaderboard, we selected three models including BPR, SVM and MLE since blending Latent Factor Model caused a loss on the public leaderboard. We further proposed a rank-based ensemble method to blend the individual models.

6. ACKNOWLEDGMENTS

This work is supported by the Natural Science Foundation of China(NSFC) under Grant No. 61003138, the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No. XDA06010202, the Innovation Fund Project of Computer Network Information Center of

Chinese Academy of Sciences "Notology-based Scientists Resource Service Platform" (Project Code: CNIC_ CX _10003) and partially supported by the National Natural Science Foundation of China (No. 61070106), the National Basic Research Program of China (No. 2012CB316300), Tsinghua National Laboratory for Information Science and Technology (TNList), Crossdiscipline Foundation and the Opening Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research (No. 5026035403). D. Bickson work is supported by grants ARO MURI W911NF0710287, ARO MURI W911NF0810242, NSF Mundo IIS-0803333, NSF Nets-NBD CNS-0721591 and ONR MURI N000140710747.

7. REFERENCES

[1] T. Anastasakos, D. Hillard, S. Kshetramade, and H. Raghavan. A collaborative filtering approach to ad recommendation using the query-ad click graph. In *International Conference on Information and Knowledge Management*, pages 1927–1930, 2009.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[3] A. P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30:1145–1159, 1997.

[4] T. Chen, Z. Zheng, Q. Lu, W. Zhang, and Y. Yu. Feature-based matrix factorization. *CoRR*, abs/1109.2271, 2011.

[5] H. Cheng and E. CantÍš-Paz. Personalized click prediction in sponsored search. In *Web Search and Data Mining*, pages 351–360, 2010.

[6] G. V. Cormack, C. L. A. Clarke, and S. Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Research and Development in Information Retrieval*, pages 758–759, 2009.

[7] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-Scale Bayesian ClickThrough rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine. In *International Conference on Machine Learning*, pages 13–20, 2010.

[8] T. Joachims. A support vector method for multivariate performance measures. In *International Conference on Machine Learning*, pages 377–384, 2005.

[9] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Knowledge Discovery and Data Mining*, pages 426–434, 2008.

[10] Y. Niu, Y. Wang, G. Sun, A. Yue, B. Dalessandro, C. Perlich, and B. Hamner. The Tencent Dataset and KDD-Cup'12. In *KDD-Cup Workshop*, 2012.

[11] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *World Wide Web Conference Series*, pages 521–530, 2007.

[12] S. Shen, B. Hu, W. Chen, and Q. Yang. Personalized Click Model through Collaborative Filtering. In *Web Search and Data Mining*, 2012.