


Predicting e-book ranking based on the implicit user feedback

Bin Cao¹  · Chenyu Hou¹ · Hongjie Peng¹ · Jing Fan¹ · Jian Yang² · Jianwei Yin³ · Shuiguang Deng³

Received: 15 January 2018 / Revised: 19 March 2018 / Accepted: 27 March 2018 /
Published online: 14 April 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract In this paper, we plan to predict a ranking on e-books by analyzing the implicit user behavior, and the goal of our work is to optimize the ranking results to be close to that of the ground truth ranking where e-books are ordered by their corresponding reader number. As far as we know, there exist little work on predicting the future e-book ranking. To this end, through analyzing various user behavior from a popular e-book reading mobile APP, we construct three groups of features that are related to e-book ranking, where some

This article belongs to the Topical Collection: *Special Issue on Deep vs. Shallow: Learning for Emerging Web-scale Data Computing and Applications*
Guest Editors: Jingkuan Song, Shuqiang Jiang, Elisa Ricci, and Zi Huang

✉ Jing Fan
fanjing@zjut.edu.cn

Bin Cao
bincao@zjut.edu.cn

Chenyu Hou
houcy@zjut.edu.cn

Hongjie Peng
phjly110@163.com

Jian Yang
jian.yang@mc.edu

Jianwei Yin
zjuyjw@zju.edu.cn

Shuiguang Deng
dengsg@zju.edu.cn

¹ Zhejiang University of Technology, HangZhou, China

² Macquaire University, Sydney, Australia

³ Zhejiang University, HangZhou, China

features are created based on the popular metrics from the e-commerce, e.g., conversion rates. Then, we firstly propose a baseline method by using the idea of learning to rank (L2R), where we train the ranking model for each e-book by taking all its past user feedback within a time interval into consideration. Then we further propose TDLR: a Time Decay based Learning to Rank method, where we separately train the ranking model on each day and combine these models by gradually decaying the importance of them over time. Through extensive experimental studies on the real-world dataset, our approach TDLR is proved to significantly improve the e-book ranking quality more than 10% when compared with the L2R method where no time decay is considered.

Keywords E-book ranking · Ranking prediction · Implicit user behavior

1 Introduction

Nowadays, due to the portability and searchability of electronic books (e-books), it is common to see people holding their mobile phones or Kindles reading e-books on the bus, train, airplane and even while standing in line. To make people aware of the popularity of e-books, e-book selling web sites like Amazon often give rankings on e-books in different viewpoints, e.g., best sellers in the day or best rated books so far. In the meantime, a lot of sellers get hooked on the ranking due to the reason that better ranking equates better exposure to the readers, which will in turn drive more traffic and sales producing more revenues over time for the seller, i.e., the higher the e-book ranks, the more profits the seller gains.

However, most existing e-book rankings are generated based on sales or other statistics over past days or weeks, which is unable to reveal the accurate future popularity of e-books. What is more important, traditional statistical rankings fail to maximize the effect of attracting more users to read more e-books since it is likely that books in the statistical ranking list had been read by many readers and their future reading quantity in terms of the number of readers will drop while some other newly issued e-books that may become popular in the future have their reading quantity increased. In order to further boost this implicit trend, we plan to predict (or we can say “recommend”) an e-book ranking, which not only disclose some future reading tendency but also maximize the total reading quantity for the e-books in the ranking list, i.e., the total number of readers in the predicted ranking is close to that of ground truth ranking. Besides, knowing future ranking for e-books in advance has many other benefits, e.g., the bookstore can stock more books that may be popular in the future based on the performance of their corresponding electronic versions; the reading interest overall could be captured by investigating the future e-book rankings; users may find more interesting e-books to read and etc.

Ranking problems have been widely studied in web search engine field, but the ranking methods from this area usually require users to provide a query, and the items within the results must have some inner connections, e.g., the content similarity between two web pages. Hence, these ranking techniques could not directly be applied to our e-book ranking prediction problem where no specific query is needed and the relations between different e-books are ignored. Another popular way to solve the problem of e-book ranking prediction is to use the idea of collaborative filtering [9], where the user-item rating matrix [13] is utilized to make top k recommendations for users. Unfortunately, ratings are not always available. To tackle this problem, implicit user feedback generated during the interaction between users and the systems have been incorporated into the ranking and recommendation and achieve

great success [2, 8]. However, these methods seldom consider the behavior changes over time which is important for us to improve the prediction quality in e-book ranking problem.

In our work, we also assume that a user's behavior to an e-book implies his preference to the e-book. Therefore, we investigate the implicit feedback from different viewpoints, namely features, and incorporate them into the e-book ranking prediction problem. It is important to note that, the goal of our work is to make the total reading quantity of predicted e-book ranking list close to that of the ground truth ranking, i.e., the actual ranking for the future. To this end, we collect a real-world dataset from a popular e-book reading mobile APP, and firstly created three groups of features based on the user reading behavior and some popular metrics that are commonly used in e-commerce, including (1) statistical features based on counting different types of behavior, (2) conversion rates which measures what happens when users are at an e-book, and (3) some ratio features that are used to capture more advanced characteristics of an e-book, such as the reader retention. Then based on these features, we propose a baseline method by adopting the idea of learning to rank (L2R) [23] to perform e-book ranking prediction, where the top-K prediction on a given date takes the role of queries in L2R and the e-books themselves can take the role of documents. In this baseline method, L2R does not take the temporal dynamics of user behavior into consideration, and some hidden trends that can influence the final ranking quality still need to be further revealed. Hence, we present TDLR: a Time Decay based Learning to Rank method, which can improve the e-book ranking quality by gradually decaying the importance of past implicit feedback over time. Specifically, TDLR firstly trains the ranking model for each book on each day, then computes a composite rating score for each e-book by assigning different weights to the ratings that derived from each learned ranking model before the prediction date, and then combining them in a linear way. The weights are determined by a carefully designed time decay function. Through extensive experimental evaluations, TDLR had been demonstrated to be more effective than pure L2R in terms of ranking quality. In general, the contributions in this paper can be summarized as follows:

1. As far as we know, we are the first to propose the problem of e-book ranking prediction, and solving this problem has many benefits for both book sellers and readers.
2. Motivated by the performance indicator in e-commerce, we designed three groups of features by studying various user reading behavior, where some of these features are used in e-book ranking domain for the first time.
3. To solve the proposed problem, instead of merely relying on one relevance criterion, e.g., the number of readers for an e-book, we firstly proposed a learning to rank based method to predict the e-book ranking by taking into account of multiple features that are related to e-book ranking, then an improved method named TDLR was present to further improve the ranking quality by decaying the importance of past implicit user feedback in the construction of ranking model.
4. Based on the metric of Normalized Discounted Cumulative Gain (NDCG), we performed extensive experimental evaluations on real-world data. The experiment results show that our TDLR method can outperform traditional learning to rank method by 10.7% when solving the e-book ranking prediction problem.

The rest of the paper is organized as follows. In Section 2, we introduce the dataset and perform data analysis to validate our motivation. The feature engineering based on the user behavior is detailed in Section 3. Then we present the method for predicting e-book ranking in Section 4. Section 5 shows the experimental results and evaluations. Related work is reviewed in Section 6. Finally in Section 7, we conclude and describe future research directions.

2 Preliminaries

In this section, we firstly overview the dataset used throughout the paper, and then we perform a data analysis to show the motivation of our work, i.e., the difference between statistical ranking and predicted ranking in terms of the reading quantity.

2.1 Dataset and preprocessing

The dataset we use is from a popular e-reading mobile app which has millions of users, and its backend log records all predefined behavior for each user since he or she starts to read with this app.

For the sake of focusing on our problem, we extracted following four fields from original logs for each user: (1) *userID*, the unique identification for the user; (2) *bookID*, the unique identification for the book; (3) *eventID*, corresponds to a kind of predefined user behavior, and (4) *timeStamp*, the time stamp that this event or behavior occurs and it is accurate to a second. For a single user, the order of these records represents his temporal behavior sequence. Note that, there were actually more than ten kinds of behavior predefined in original log system. But, due to the reason that some behavior is similar or lacks of corresponding records, we finally choose following six kinds of reading behavior in our work: (1) add the e-book to the reading list, (2) remove the e-book from the reading list, (3) download the e-book, (4) click the e-book cover, (5) turn the page, and (6) click the e-book catalog.

We will use this preprocessed data set for analysis in the rest of this paper and for privacy and contract reasons, we hide some confidential information which will not prevent readers from understanding the data itself, such as user name and book name (we use their IDs instead).

2.2 Data analysis

Next, we perform data analysis to demonstrate the motivation of our work by answering following two questions: (1) Do e-book ranking results change frequently over time? (2) How ineffective the statistical ranking is?

2.2.1 Daily changes of e-book ranking

To answer the first question, we perform two tasks for two consecutive daily e-book top-k rankings by calculating the probabilities of the top ranked e-books that: (1) only appear in the ranking list of the second day; and (2) appear in both two days but their orders in the ranking list are changed. To this end, we sort the e-books in descending order by the number of corresponding readers on each day, and then a daily top-k ranking could be derived. Here we simply take $k=10$ for observation.

Through observing a whole month's daily top-10 rankings, we find that the proportion of e-books that newly appear in top-10 ranking on the second day of any two consecutive days is 19%, i.e., 1.9 e-books in average which were not ranked among top-10 on first day and they will jump to the top-10 on the next day. Furthermore, we also discover that 72% of e-books in average have their orders changed when compared with their corresponding orders one day before, which is a quite different case. Hence, from above observations, we can infer that e-book ranking varies much as time goes by and traditional ranking based on past statistics cannot reflect the future ranking accurately.

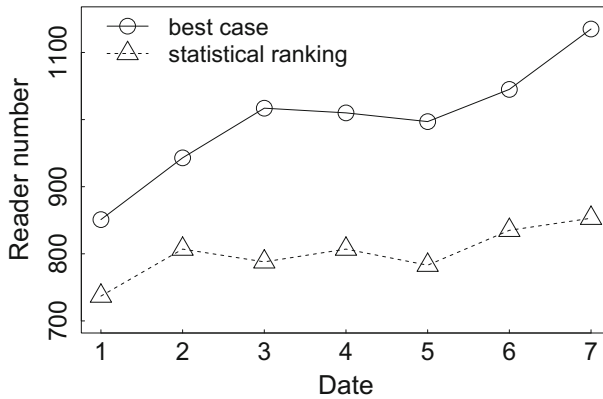


Figure 1 Using statistical ranking for prediction

2.2.2 Pitfall of statistical ranking

For the second question, considering that the real e-book ranking on the $(i+1)$ th day can only be captured by the end of the day, we use the statistics collected from past i days to derive a top-10 ranking as a prediction for the $(i+1)$ th day, and compare it with the actual top-10 which is made based on the statistics on the $(i+1)$ th day. The difference of above two ranking methods is shown in Figure 1, where the dotted line represents the total number of readers (Y-axis) for top 10 e-books, which were selected by using the statistics from the past week, on different days (X-axis), while the reader number for actual top-10 e-book ranking on each day is plotted in solid line.

As we can see from Figure 1, there exists a big gap between two rankings for all days. For example, on day 4, the total number of readers for the top-10 e-books listed in the actual ranking is 25% more than that of top-10 e-books from the statistical ranking derived from past days. The main reason behind this is that some inner reading trends have been concealed by statistics. Next, we dig into more details of our data for further explaining why statistical ranking is ineffective. In Table 1, each column shows the daily reader number for three books (e-book names are anonymous with no impact on our work). By using statistics over past 4 days (Sum column), i.e., day($i-4$) to day ($i-1$), we can get the ranking of “B1,B3,B2” (based on $359 > 209 > 156$). However, we can clearly see that e-book B2 has an increasing reader number as approaching day (i), while B1 shows a decreasing trend and B3 is relatively stable. So, intuitively, B2 should be ranked higher and in fact, the actual ranking on day (i) as shown in Table 1 should be “B2,B3,B1” (due to $80 > 48 > 29$).

Table 1 Inner reading trends hidden by statistics

e-book	Day ($i-4$)	Day ($i-3$)	Day ($i-2$)	Day ($i-1$)	Sum	Day (i)
B1	142	98	77	42	359	29
B2	13	30	42	71	156	80
B3	51	44	62	52	209	48

Last but not least, higher rank for an e-book can in turn bring more traffic or profits for its providers, which is also ignored by the statistical ranking based on the above analysis. Hence, the rest of this paper targets to design a more effective e-book ranking method to reveal the above hidden trends from the historical user reading behavior and minimize the reading quantity (in terms of the reader number) gap between the predicted and the ground truth top-k rankings.

3 Feature engineering

In this section, motivated by performance metrics that are widely used in e-commerce, we present the features that may have impact on e-book ranking prediction. These features are categorized into three groups according to the characteristics they try to capture for a given e-book, i.e., (1) *statistical features*, such as the number of downloads, (2) *conversion rates*, such as the proportion of readers who take action to read the e-book, and (3) *ratio based features*, which refer to more advanced features that are created based on former two categories, e.g., rate of return within a week. Note that, though the e-book ranking prediction can be made per hour or per day, we plan to update the e-book ranking daily in our work since it is more common in real-world. Hence, the statistical features and conversion rates features are extracted from the original user behavior by the day. But the ratio based features are extracted by spanning a couple of days.

Statistical Features Usually, time and frequency are two important types of user behavior. For example, dwell time on items have been converted into user-rating matrix for helping improve recommendation performance [31], and the frequency at which users access their online social networks has been well studied since frequent repeated activities naturally serve as good targets for advertisements [3]. However, there was no such feedback in original data and it is far from trivial to create them. Next, based on the original behavior recorded, we mainly introduce how we create the features of the time spent for reading an e-book and the frequency a user reads the same book. Then the rest features could be easily built based on them.

First of all, we need to compute the user dwell time on a single page before next page turning occurs. To do this, we retrieve the data that contains the same behavior of turning the page, and then we can get the dwell time on a single page of one e-book by calculating the time stamp difference between two adjacent records. For example, there are two adjacent records of page turning behavior for the same user on an e-book, and their *timeStamp* values are “017/07/04 13:27:05” and “2017/07/04 13:27:09”, the difference of them is 4 seconds which will be considered as the time spent on a single page.

However, we find there are some unusual cases where the difference could be very huge, e.g., hours or days, which will not be regarded as the dwell time for a single page since its corresponding adjacent records actually represent different reading behavior in different periods. Hence, we need to further find out the range of reasonable dwell time for a common person to read a single page. By analyzing the statistics of each book page for all users, as shown in Figure 2, we find that 80% page dwell time is less than 300 seconds. According to Pareto theorem, we will regard dwell time that exceeds 300s as the outlier, i.e., two different reading behaviors, and preclude them from consideration. Similarly, page dwell time that is lower than 2s is also ignored, since it maybe caused by user quickly turning the page that has been read before and this is not the normal reading behavior.

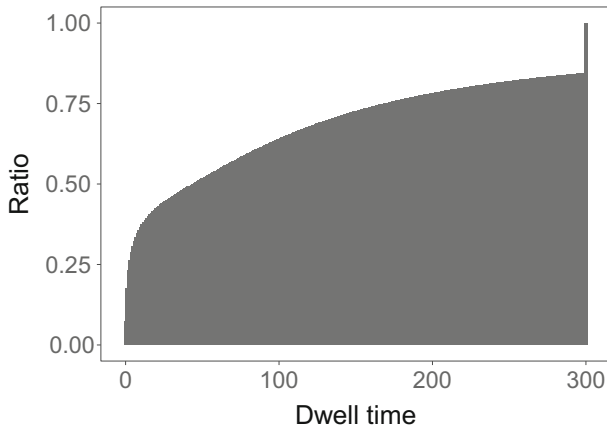


Figure 2 Statistics for dwell time on a single page

As a result of above processing, page dwell time between “(2s, 300s)” is chosen for evaluation. For the time difference that is over 300s, we will mark corresponding records as reading the e-book twice, i.e., the user reads this e-book in different periods for twice. In this way, we can derive the reading frequency for each user to each e-book he reads. Moreover, the total reading time spent on an e-book for a user can be calculated by accumulating each page dwell time of the book. Next, based on the aforementioned two types of user behavior, i.e., dwell time and reading frequency, we can further create more statistical features for each e-book. As shown in Table 2, the feature names and their descriptions are given, and they are self-explanatory.

Conversion Rates Features The conversion rate is the proportion of conversions among total number of visitors for a website or an app, where a conversion refers to any desired action that managers want the user to take. Conversion rates have been one of the most popular and important tools in e-commerce [12] to gauge the success of websites or apps and identify areas for improvement. This motivated us to test them in the domain of e-book ranking. The general intuition behind them is that the conversion rate can not only

Table 2 Statistical features

Feature	Description
<i>s-arlist</i>	n. of times added to the reading list
<i>s-rrlist</i>	n. of times removed from the reading list
<i>s-downl</i>	n. of downloads
<i>s-bcc</i>	n. times the book cover had been clicked
<i>s-rq</i>	reading quantity (i.e., overall reading frequencies)
<i>s-acc</i>	the amount of catalog clicks
<i>s-prb</i>	n. of people that have read the book
<i>s-trt</i>	the total reading time
<i>s-pabr</i>	n. of people that have added the book to the reading list
<i>s-prbr</i>	n. of people that have removed the book from the reading list
<i>s-pdb</i>	n. of people that have downloaded the book

Table 3 Conversion rates features

Feature	Description (Symbols ‘%c’ stands for the conversion rate)
<i>c-qr</i>	%c of reading quantity: <i>s-rl/s-prb</i>
<i>c-tr</i>	%c of spent reading time: <i>s-trt/s-prb</i>
<i>c-par</i>	%c of adding to the reading list: <i>s-pabr/s-prb</i>
<i>c-prr</i>	%c of removing from the reading list: <i>s-prbr/s-prb</i>
<i>c-pdr</i>	%c of downloads: <i>s-pdb/s-prb</i>

indicate some level of engagement with the e-book but also can track for assessing whether the corresponding user feedback works for e-book ranking. We use five conversion rates features which are shown in Table 3, and this is the first time for these features to be used in e-book ranking prediction.

Ratio based Features In order to further capture as many as possible characteristics for e-book ranking, we also investigate some more advanced performance metrics that are popular used in e-commerce, e.g., the rate of return and user retention. Drawing on the experience of the successful use of these advanced features, we also design three similar features that can intuitively reflect how well the given e-book can retain its readers:

- *Rate of return*, which is the number of users that read the given e-book repeatedly within one week divided by the total number of readers for the e-book within the same duration.
- *Reader retention*, which refers to the number of readers that read the e-book for consecutive two days divided by the number of the readers that read the e-book in the second day.
- *Bounce rate*, which is the percentage of readers to a given e-book who has only read the e-book once during one week.

Unlike former two groups of daily features, these features are created by spanning a couple of days. As far as we know, we are the first to test them in e-book ranking domain.

Feature Normalization Since three groups of features have different scales, e.g., ratio based features vary from 0 to 1 while statistical features are often larger than 1. Therefore, it is necessary to normalize these features so that our model can work properly. The method of normalization is feature scaling where we rescale each feature to the range in [0, 1]. The formula is given as follows:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (1)$$

where x is the original value of a sample, x' is the normalized value of the sample, and $\min(x)$ is the minimal value of the feature while $\max(x)$ is the maximum.

4 Predicting e-book ranking

This section presents two techniques to solve the problem of e-book ranking prediction. The first baseline method utilizes the common machine learning based ranking idea, i.e., learning to rank (L2R) [23], on a pair of ranking days, while the second one considers the temporal character of user feedback and combines a time decay function with the learning to rank model to improve the ranking results.

4.1 Predicting with learning to rank

In our problem, the optimal ordering of e-books could be judged by the number of readers for the top-K ranked e-books, i.e., the more readers the better. However, as demonstrated in Section 2.2, merely relying on the statistics for historical user behavior can not help us to achieve the goal since the future popularity of e-books has many signals that are hidden in the user behavior. Hence, it is natural to treat the construction of e-book ranking model as a supervised machine learning problem, and then the idea of learning to rank (L2R) can be a good start.

In L2R, each query-document pair consists of a set of features of a document and a numerical or ordinal score denoting how relevant the document is to a query. Then a set of such pairs will be used to train a machine learning model to predict the relevance of other documents that haven't been scored. To apply the idea of L2R to our e-book ranking prediction problem, the role of queries in L2R can then be taken by top-K ranking prediction on someday and the role of documents can be taken by e-books. Specifically, each e-book (f, r) to be ranked can be represented by a feature vector f consisting of the created features in Section 3, and an ordinal score r describing its relevance rating, i.e., higher score indicates higher rank. In our work, the training data for L2R consists of two parts where the feature vector f of an e-book is the average of n days before day(i) and its corresponding ranking score r is from day(i). In the meantime, the validation data is composed of e-book features from day(i) and the e-book ordering based on the reader number from day(i+1), i.e., the day to be predicated. Based on this, we can perform L2R on the training data before day(i) for learning the ranking model by using any supervised machine learning classification method, e.g., Random Forest [4], and then apply this model to the e-books of day(i) to generate a predicted rating score for each e-book, which can be used as a comprehensive rating criterion to generate the final ranking list for day(i+1).

It is important to note that since we feed the learned model with a set of e-books D_i on day(i), the predicted e-book ranking list RL_{i+1} on day(i+1) can only contain the e-books that had feedback on day(i), i.e., $RL_{i+1} \subset D_i$. Here a question may arise: *Is it still meaningful to do the prediction for future even if there will be no new e-book shown in the list?* The answer is yes: for an e-book that had not been read by any one in the first day, the odds of jumping up into the top-K rankings are very small, and it is reasonable for us to exclude this abnormal case from consideration.

4.2 TDLR: time decay based learning to rank

In above pure L2R based baseline method, since the temporal dynamics of user behavior are ignored, some trends for implicit feedback over time are hidden and intuitively, it is possible for us to further improve the ranking quality by taking them into account. To this end, we present our second solution - TDLR: a Time Decay based Learning to Rank method. TDLR can achieve the improvement on ranking by gradually decaying the importance of past user behavior as time goes by. Next, we detail TDLR from two aspects: the design of time decay function and TDLR based ranking.

4.2.1 Time decay function design

Intuitively, the user feedback that occurred long time ago has less impact on the future e-book ranking prediction than those happened in a shorter time, i.e., the behavior influence will decay over time. However, the way how time decay varies. Usually, there are three main

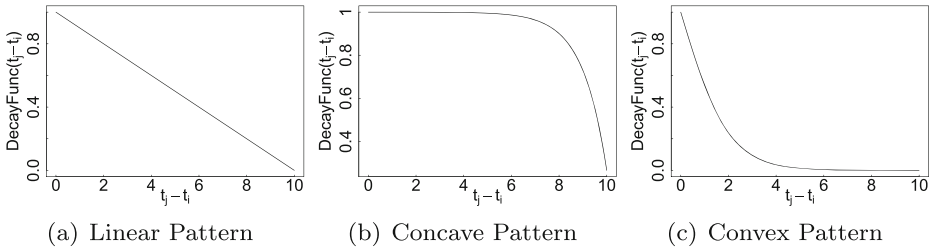


Figure 3 Time decay patterns

patterns: linear, concave and convex. As illustrated in Figure 3, linear pattern is the simplest form to model a constant rate of decay, while the concave and convex decay functions are just two opposite patterns for describing how the decay speed grows as time goes by, e.g., first slow then fast for concave (Figure 3b). Psychologically, the behavior that occurs closer to the prediction date has bigger influence. Hence, in our work, we choose the convex pattern, where the decay initially decreases ver slowly then speeds up (Figure 3c), as the time decay model for user behavior.

Specifically, we adopt a variant of Sigmoid function for our problem. As shown in (2), $|t_j - t_i|$ represents the interval of two different dates t_j and t_i . For example, suppose we are going to predict the e-book ranking on July 13th (t_j), and there is a behavior model had been trained on July 11th (t_i), then we can apply the interval of these two dates, i.e., 2 days, to (2) and get the decayed importance of the model learned on July 11th. Note that, due to the numerator 2 in (2), the importance range derived from it is (0, 1] where “0” indicates the model can contribute none to the prediction while “1” denotes the greatest importance. In addition, α is used to control the slope of decay curve shown in Figure 3c, i.e., bigger α indicates faster decay speed in the beginning, and its corresponding curve becomes steeper as well.

$$DecayFunc(|t_j - t_i|) = \frac{2}{1 + e^{\alpha|t_j - t_i|}} \quad \alpha > 0 \tag{2}$$

4.2.2 TDLR based ranking

The main idea of TDLR is to combine the time decay function with learning to rank to perform e-book ranking prediction by incorporating the implicit user feedback which is influenced by the temporal factor. Note that, unlike pure L2R method in Section 4.1 where only a whole ranking model was learned over n days before day(i) (suppose day(i+1) is the day to be predicted), TDLR learns ranking model for each e-book on each day before day(i), i.e., for an e-book, there will be different ranking models on different days. Based on the set of learned models, the workflow of TDLR ranking prediction for day(i+1) is demonstrated in Figure 4.

Suppose we are going to predict the e-book ranking on day(i+1), as shown in Figure 4, TDLR takes as the input the dataset of day(i), denoted by D_i , and an integer number n indicating the number of days to be considered for e-book ranking prediction. The output of TDLR is a relevance rating $r(c, w)$ with the range of (0, 1) for each e-book in D_i . $r(c, w)$ will be viewed as the ordering basis in our work. Within TDLR, D_i is fed into each learned model of n days before day(i+1), i.e., $Model_{i-n+1}$ to $Model_i$, and then a corresponding group of intermediate relevance scores $\{c_1, c_2, \dots, c_n\}$ for each e-book is derived. Here, each score is also in the range of (0, 1) and we consider that they have some impact on the final

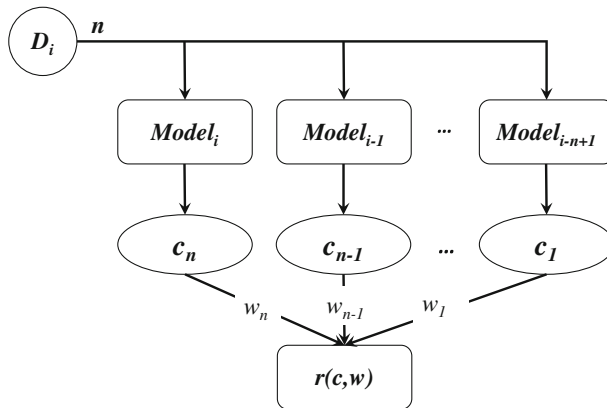


Figure 4 Workflow of TDLR

relevance rating $r(c, w)$. In the end, the final relevance rating $r(c, w)$ can be computed by using (3), where the weight vector $\{w_1, w_2, \dots, w_n\}$ is generated based on the time decay function $DecayFunc$ and hence, each $w_j (1 \leq j \leq n)$ represents the importance of the relevance score of c_i that is generated based on the model of day(i).

$$r(c, w) = w_n c_n + w_{n-1} c_{n-1} + \dots + w_1 c_1 \tag{3}$$

Since the final relevance rating $r(c, w)$ is defined to be between 0 and 1, we can also view it as the possibility to be ordered to the top. To make the range of (3) between 0 and 1, we only need to make the sum of $\{w_1, w_2, \dots, w_n\}$ equal to 1 due to that each relevance score c derived from the model on each day is in the range of (0, 1). Hence, we normalize the decayed value for each c_i with (4).

$$w_i = \frac{DecayFunc(i)}{\sum_{j=1}^n DecayFunc(j)} \tag{4}$$

5 Experimental evaluation

In this section, based on the real log from a popular e-book reading mobile APP described in Section 2, we perform detailed experimental evaluations from following aspects: (1) Feature analysis, where we study different impact from three groups of features; (2) Effect of time decay, where we study how the time decay function described in Section 4.2.1 influence the ranking results; (3) Comparison study, where we compare our TDLR against traditional learning to rank methods without time decay and other TDLR variants equipped with different classification methods. In our experiments, we totally collected 30 consecutive days of log data from July 1st to July 30th 2013, i.e., a whole month, and we only consider e-books that have been read for more than 20 days and active users who have feedback no less than 20 days. Hence, after cleaning the collected data by removing inactive users and abnormal records, 10,667 users and 14,260 e-books in total are left for analysis. Based on this, we split the data to five parts and conduct 5-fold cross validation. For each fold, as mentioned in Section 4.1, the test dataset that contains the e-book to be ranked and their ground truth rankings is from day(i-1) and day(i), while the training dataset contains 10 consecutive days before the prediction date day(i).

The Learning to Rank strategy we used is point-wise which essentially looks at a single document at a time in the loss function. Due to that a point-wise L2R using Random Forests (RF) has been shown relatively effective in several real world benchmarks [20], we also choose this method as our main ranking strategy in our experiments (and without specific note, TDLR is integrated with RF by default). The implementation of RF and other machine learning methods used in this paper is provided by Scikit-learn [22] in Python, and without specific explanation, the parameter settings for those methods are used by default.

For evaluation methodology, we exploit the metric of *Normalized Discounted Cumulative Gain at top K* (NDCG@K) [29]. This metric is the most popular measure of ranking quality and it takes the position of an e-book in the consideration of the usefulness of ranking. NDCG is the normalized value of DCG which is formally defined as follows:

$$DCG@K = \sum_{i=1}^K \frac{2^{r_i} - 1}{\log_2(i + 1)} \tag{5}$$

where r_i represents the relevance values of documents in IR. Here, we take the role of r_i as the true rating score for the e-book at position i in the predicted ranking. In our experiments, the true rating score r_i is given by (6):

$$r_i = K - i + 1 \tag{6}$$

where i is the true ranking position, and K is the number of e-books to be ranked. For example, for a top-K ($K=10$) e-book ranking task, the first ranked e-book in the ground

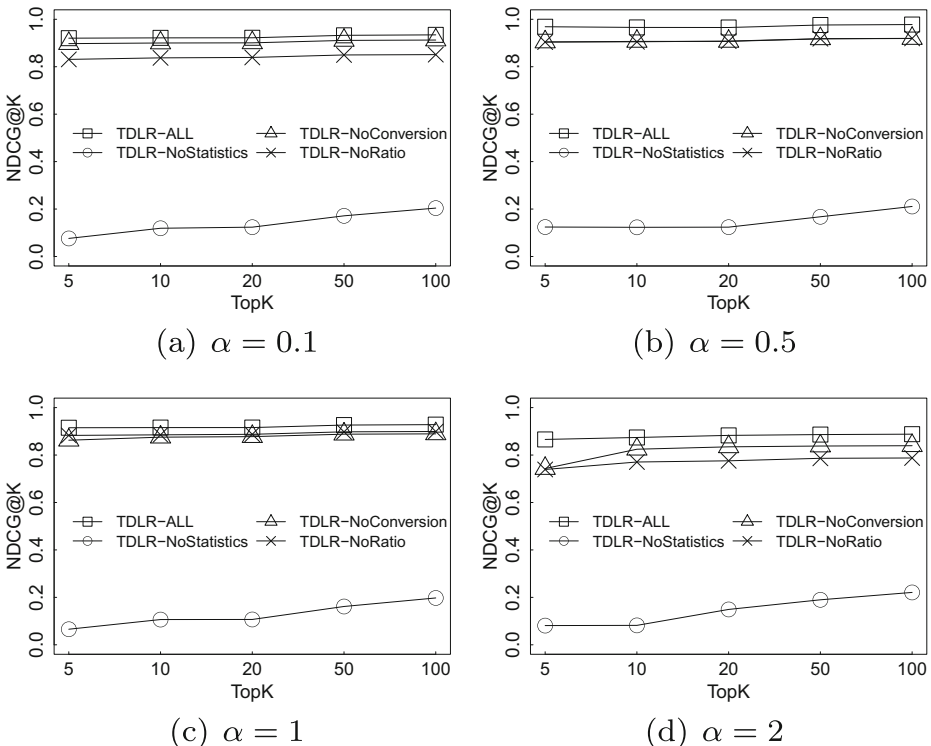


Figure 5 Feature Analysis in terms of NDCG@K v.s. α (decay day = 6)

truth ranking will be assigned a rating score of 10 ($10 - 1 + 1$) while the tenth book has a rating score of 1. An ideal DCG for top-K (IDCG@K) ranking is defined as the DCG@K of the ground truth ranking, then NDCG@K can be calculated by (7).

$$NDCG@K = \frac{DCG@K}{IDCG@K} \tag{7}$$

5.1 Feature analysis

To analyze the impact of each group of features, we compare following four alternatives of TDLR: (1) TDLR-ALL, where we take all features into consideration in the construction of ranking model; (2) TDLR-NoStatistics, where we exclude the statistical features; (3) TDLR-NoConversion, where no conversion rate feature is considered; and (4) TDLR-NoRatio, where the ratio based features are excluded. Figures 5 and 6 both show how above four TDLR alternatives behave in terms of NDCG@K, the difference between them is the parameter setting for the time decay function, i.e., α and decay day $|t_j - t_i|$ of (2). Specifically, we perform four tests by using different α values while fix $|t_j - t_i|$ to 6 in Figures 5, and 6 contains tests under different decay day with fixed $\alpha = 0.1$. From two figures, we can get following observations.

As we can see from Figures 5 and 6 TDLR-ALL outperforms other three alternatives in all cases, which implies that all features we designed contribute to the improvement of NDCG@K. Specifically, TDLR-NoConversion and TDLR-NoRatio have little less

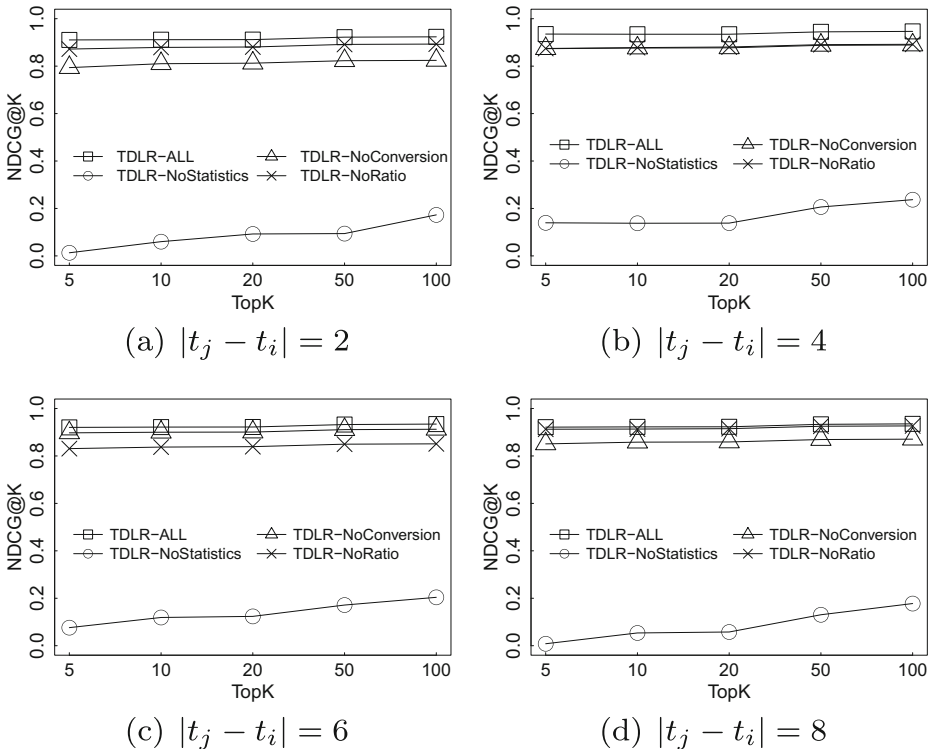


Figure 6 Feature Analysis in terms of NDCG@K v.s. decay day ($\alpha = 0.1$)

NDCG@K than TDLR-ALL. However, compared with each other, they have a trend of fluctuation, i.e., sometimes TDLR-NoConversion outperforms TDLR-NoRatio and sometimes vice versa. It means that conversion rates features and ratio-based features are indeed helpful to improve the quality of rank prediction, but the contribution of them somehow varies at different cases so that we can not tell the importance of them exactly. In addition, there is a surprising observation that the NDCG@K of TDLR-NoStatistics is significantly lower than TDLR-ALL. It indicates that statistical features contribute the most for prediction. The reason behind this is that statistical features are good at capturing how attractive the e-books are to readers. For example, *prb* represents the number of readers for e-books. It directly reflects the attraction of books. The more readers is, the more attractive an e-book is. Besides, we will rank e-books by mistake if we merely consider conversion rates features and ratio-based features. For example, there are two e-books that one of them has been read by one reader while the other one has been read by 1,000 readers, but it is possible that they have the same value of *c-qr*. We can not distinguish them by *c-qr*, which may results in wrong rank prediction result. This phenomenon tells us that in our practical scenarios, although these statistical features such as *prb* are easy to get, they are the most useful to improve the rank prediction.

Table 4 Time decay effect result

	Decay day	1	2	3	4	5
k=10	$\alpha = 0.1$	0.8617	0.9115	0.9561	0.9345	0.9607
	$\alpha = 0.5$	0.8617	0.9114	0.9348	0.9359	0.9614
	$\alpha = 1$	0.8617	0.9109	0.9115	0.9114	0.9157
	$\alpha = 2$	0.8617	0.8617	0.8741	0.8741	0.8741
k=50	$\alpha = 0.1$	0.8747	0.9222	0.9663	0.9446	0.9707
	$\alpha = 0.5$	0.8747	0.9224	0.9452	0.9461	0.9717
	$\alpha = 1$	0.8747	0.9219	0.9223	0.9221	0.9264
	$\alpha = 2$	0.8747	0.8864	0.8863	0.8863	0.8863
k=100	$\alpha = 0.1$	0.8763	0.9236	0.9682	0.9466	0.9726
	$\alpha = 0.5$	0.8763	0.9237	0.9467	0.9481	0.9736
	$\alpha = 1$	0.8763	0.9233	0.9237	0.9239	0.9282
	$\alpha = 2$	0.8763	0.8880	0.8880	0.8880	0.8880
	Decay day	6	7	8	9	10
k=10	$\alpha = 0.1$	0.9217	0.9212	0.9226	0.9228	0.9229
	$\alpha = 0.5$	0.9658	0.9657	0.9657	0.9657	0.9657
	$\alpha = 1$	0.9158	0.9158	0.9158	0.9158	0.9158
	$\alpha = 2$	0.8741	0.8741	0.8741	0.8741	0.8741
k=50	$\alpha = 0.1$	0.9325	0.9319	0.9333	0.9336	0.9337
	$\alpha = 0.5$	0.9760	0.9758	0.9757	0.9757	0.9757
	$\alpha = 1$	0.9264	0.9264	0.9264	0.9264	0.9264
	$\alpha = 2$	0.8863	0.8863	0.8863	0.8863	0.8863
k=100	$\alpha = 0.1$	0.9344	0.9338	0.9353	0.9355	0.9356
	$\alpha = 0.5$	0.9779	0.9776	0.9776	0.9776	0.9776
	$\alpha = 1$	0.9283	0.9283	0.9283	0.9283	0.9283
	$\alpha = 2$	0.8880	0.8880	0.8880	0.8880	0.8880

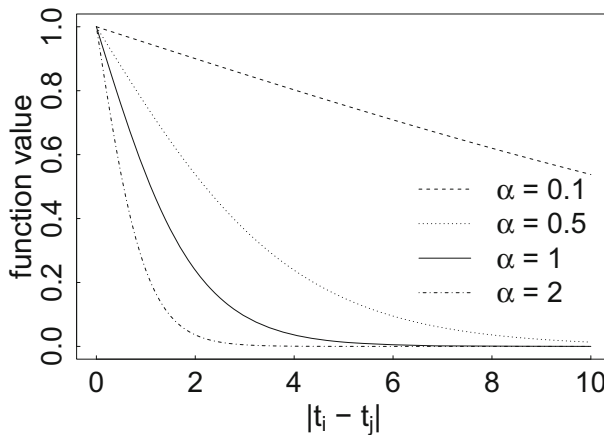


Figure 7 Decay function

Based on above evaluation results, for the rest experiments, we take all features into consideration for ranking model training.

5.2 Time decay effect study

In TDLR model, $|t_j - t_i|$ determines how many days of learned ranking models it would take into account for decaying, while α controls how fast the importance of each model decay over time. We now look forward to evaluating the relation between these two important parameters and determine the best parameter settings to optimize the ranking quality of TDLR. To this end, we randomly choose five days for testing, they are July 20, July 22, July 24, July 26 and July 28 respectively. And for each day, we divide the test to three cases by varying the number of e-books to be ranked, i.e., K . Table 4 shows the detailed result under different α and *decay day* settings. Two main observations can be derived from it.

Firstly, the most important thing is that TDLR model has the best performance when $\alpha = 0.5$ and $|t_j - t_i|$ is 6 at all cases. It is necessary to state that the values of NDCG@K when *decay day* = 1 are the same under different α . The reason behind this is that when *decay day* = 1, we actually only utilize the model from the previous day and the weight of it is 1.

Secondly, TDLR models with different values of α have different improvement. As we can see from the Table 4, NDCG@K of the TDLR with $\alpha = 2$ grew slowly when *decay day* increased while TDLR with other α grew significantly when *decay day* = 2. It can be explained by the reason that the TDLR with $\alpha = 2$ attaches less importance to historical data. The Figure 7 shows the graphic representation of decay functions with different α . We can see that the value of decay function with $\alpha = 2$ decays very fast. Therefore, the

Table 5 Time decay effect conclusion

Parameter	Range	Optimal setting
α	0.1 ~ 2	0.5
Decay day	1 ~ 10	6

weights of historical models are very small, which results in little influence on the final TDLR model. What’s more, we can see that TDLR model with a certain α will converge to a stable value when the value of *decay day* exceeds 6. It means that the models of long time ago are useless to improve the quality of rank prediction. In practice, it is reasonable to consider the models of past few days. Finally, based on the results shown in Table 4, we conclude the values of α and decay day, as shown in Table 5. Empirically, we use $\alpha = 0.5$ and decay day = 6 in following experiments to optimize the ranking quality of TDLR.

5.3 Comparison study

In this section, we perform comparison studies by answering following two questions: (1) How well TDLR performs when compared with the baseline method of L2R (denoted by Pure-L2R) and the statistical method? (2) How TDLR works for different supervised machine learning methods? To answer the second question, besides Random Forest, we also integrate TDLR with other popular classification algorithms, e.g., Support Vector Machine (SVM) [7], Bay-es [6], and Gradient Boost Decision Tree (GBDT). Note that, all features are used in the construction of ranking models for all the methods, and all TDLR based methods use the parameter setting of $\alpha = 0.5$ and $|t_j - t_i| = 6$.

Figure 8a plots the NDCG@K of statistical method, Pure-L2R and TDLR. For the sake of equality, we take previous six days data into consideration for Pure-L2R method. We can see that the NDCG@K of Pure-L2R and TDLR are both better than that of the statistical method, and meanwhile TDLR has the best performance in all cases. Specifically, TDLR outperforms the statistical method and Pure-L2R by 13.3% and 10.7% respectively. Hence, TDLR can predict better ranking results than other two methods.

Figure 8b shows the NDCG@K of different TDLR variants which are equipped with different classification methods. Here, we explicitly use TDLR-RF to represent the TDLR with Random Forest. Obviously, TDLR-RF is the best choice among all the methods for e-book ranking prediction, and this is due to the reason that random forests is an ensemble learning method for classification [35], which can use multiple decision trees to obtain better predictive performance. Since detailed technical explanations for this comparison are beyond our concern, we will not discuss them more here.

Finally, according to above experiments, the TDLR method proposed in our work outperforms other baseline methods and can achieve the best performance when predicting the e-book ranking.

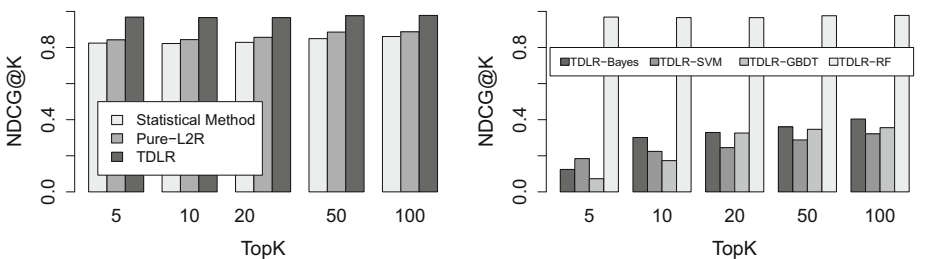


Figure 8 Model comparison

6 Related work

As far as we know, there is little work directly related to e-book ranking prediction. However, ranking on other items have been studied in a variety of other research areas and applications, such as QoS ranking [18, 34] in service computing, ranking the given answers in Q&A forums [10, 28], ranking spatial keyword queries [33] and scientific paper ranking [15]. Specifically, Zheng et al. [34] proposed a QoS ranking prediction framework by utilizing the past service usage experiences of consumers. They proposed a personalized ranking prediction framework and predicted QoS ranking for each individual. However, in our problem, we try to rank e-books for all readers rather than individuals. Dalip et al. [8] tried to rank answers for a Q&A forum by using L2R with random forest. They extracted eight groups of features from several aspects and used random forest to learn a ranking model and then predict answers for each question. Though they use L2R like we do, there is no temporal factor considered in their method. Hassan et al. solved the problem of ranking prediction on scientific articles [25] where they argue that the future references is more useful than the past citations, and based on the citation of different papers, some link relations could be created in their problem and then PageRank [21] based method is adopted to solve their problem. However, PageRank cannot be applied in our work since there is no such link between two e-books in our problem setting. Compared with above work, the research object we are facing is totally different, hence the features must be designed and created from scratch, which can directly distinguish our work from them.

Since the essence of our problem is ranking, the methods used for solving ranking problem are also related to our work. Basically, there are two main categories: (1) recommendation systems, which refer to inferring users' personal preference [27] and recommending special items to users [26]. (2) information retrieval, where web search engines retrieve documents that are relevant to queries [33].

For the first category of recommendation systems, a widely used method to recommend is collaborative filtering [9, 16, 24], which builds a model according to each user's rating to a set of items. Then this model is used to predict items that a user may be interested in. However, it is difficult to obtain users' rating in many applications. Therefore, there is a second method to recommend, named content-based filtering. Content-based filtering are based on several keywords of one item and a profile of the user's preferences [1, 19]. This method tries to recommend items that are similar to those that a user like before by comparing keywords. However, in our problem, there is neither users' rating nor keywords of books. Though there exists some work utilizing implicit feedback from users to infer users' preference [14, 30, 31], they focus on different targets or merely one feature, e.g., dwell time. While in our work, we not only consider different types of user feedback but also investigate the temporal dynamics of them, which haven't been well studied in this category of work.

For the second category of information retrieval, ranking is the central part of many problems in this field. Usually, computing numerical scores for query/document pairs with cosine similarity or other similarity measurements is the first step for ranking. However, traditional similarity methods focus on one criterion which is not reliable in many scenarios. Hence, a machine learning based ranking method, named *learning to rank* (L2R) was proposed. Unlike the collaborative filtering, L2R often utilizes some implicit features to learn a rank model. For example, Joachims T. presents an approach to learn retrieval functions and optimizing the retrieval quality of search engines using clickthrough data [11]. Y. Lv et al. utilized L2R to identify a list of related news and recommend to a user after he or she has read a current news article [17]. Yu et al. [32] presented a method incorporating visual

features and click features to learn to rank for image retrieval. Cao et al. [5] firstly proposed list-wise approach to learning to rank which aimed to improve the ranking quality of L2R. We also use L2R idea in our work, but we equip L2R with time decay function and make it even more powerful for ranking, and this can distinguish our work from the work in this category.

In summary, the problem we want to solve in this paper could be distinguished from previous research from two main aspects: (1) as far as we know, there is little work had been done on the problem of e-book ranking prediction, the features of e-book reading have not been thoroughly studied before; (2) to the best of our knowledge, we are the first to integrate time decay with L2R method when perform ranking tasks.

7 Conclusions

In this paper, we proposed the e-book ranking problem which hasn't been well addressed so far. In order to solve this problem, we firstly extracted three groups of features, named statistical features, conversion rates features and ratio based features. Then we proposed two methods based on the idea of learning to rank (L2R). In the first baseline method, we directly utilized above features to train a ranking model. To further improve the ranking quality, we then presented the second method of TDLR: a Time Decay based Learning to Rank, where a time decay function is integrated to gradually decrease the importance of past feedback over time. According to the experiments, we obtained following conclusions: (1) All features we extracted work well in our problem ; (2) TDLR method outperforms pure L2R method by 10.7% average in terms of NDCG@K; (3) TDLR method using Random Forest is the best choice for our problem.

As future work, we are going to further study the users' behavior on reading e-book and explore more reliable features to improve the quality of ranking. Moreover, we intend to inspect more insights on the relation between the ranking quality and temporal behavior.

Acknowledgements This research was partially supported by National Natural Science Foundation of China (No. 61602411, No.61772459, No.61772461), National Key Research and Development Program of China(No.2017YFB1400603), Key Research and Development Project of Zhejiang Province (No. 2015C01034, No.2015C01027), Natural Science Foundation of Zhejiang Province (No.LR18F020003).

References

1. Aggarwal, C.C.: Recommender systems: the textbook. Springer Publishing Company, Incorporated, Berlin (2016)
2. Agichtein, E., Brill, E., Dumais, S.: Improving Web search ranking by incorporating user behavior information. In: SIGIR'06, pp. 19–26. ACM (2006)
3. Benevenuto, F., Rodrigues, T., Cha, M., Almeida, V.: Characterizing user behavior in online social networks. In: IMC'09, pp. 49–62. ACM (2009)
4. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)
5. Cao, Z., Qin, T., Liu, T.Y., Tsai, M.F., Li, H.: Learning to rank: from pairwise approach to listwise approach. In: ICML'07, pp. 129–136. ACM (2007)
6. Chib, S.: Bayes regression with autoregressive errors: a gibbs sampling approach. *J. Econ.* **58**(3), 275–294 (1993)
7. Cortes, C., Vapnik, V.: Support vector machine. *Mach. Learn.* **20**(3), 273–297 (1995)
8. Dalip, D.H., Gonçalves, M.A., Cristo, M., Calado, P.: Exploiting user feedback to learn to rank answers in Q&A forums: a case study with stack overflow. In: SIGIR'13, pp. 543–552. ACM (2013)

9. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* **22**(1), 5–53 (2004)
10. Jeon, J., Croft, W.B., Lee, J.H., Park, S.: A framework to predict the quality of answers with non-textual features. In: *SIGIR'06*, pp. 228–235. ACM (2006)
11. Joachims, T.: Optimizing search engines using clickthrough data. In: *KDD'02*, pp. 133–142. ACM (2002)
12. Kohavi, R.: Mining e-commerce data: the good, the bad, and the ugly. In: *KDD'01*, pp. 8–13. ACM (2001)
13. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* **42**(8), 30–37 (2009)
14. Lee, D.H., Brusilovsky, P.: Reinforcing recommendation using implicit negative feedback. In: *International conference on user modeling, adaptation, and personalization*, pp. 422–427. Springer (2009)
15. Lehmann, S., Jackson, A.D., Lautrup, B.E.: Measures and mismeasures of scientific quality arXiv preprint physics/0512238 (2005)
16. Liu, J., Tang, M., Zheng, Z., Liu, X.F., Lyu, S.: Location-aware and personalized collaborative filtering for web service recommendation. *IEEE Trans. Serv. Comput.* **9**(5), 686–699 (2016)
17. Lv, Y., Moon, T., Kolari, P., Zheng, Z., Wang, X., Chang, Y.: Learning to model relatedness for news recommendation. In: *WWW'11*, pp. 57–66 (2011)
18. Mao, C., Chen, J., Towey, D., Chen, J., Xie, X.: Search-based qos ranking prediction for web services in cloud environments. *Futur. Gener. Comput. Syst.* **50**, 111–126 (2015)
19. Melo, E.V.D., Nogueira, E.A., Guliato, D.: Content-Based filtering enhanced by human visual attention applied to clothing recommendation. In: *ICTAI'15*, pp. 644–651 (2015)
20. Mohan, A., Chen, Z., Weinberger, K.: Web-Search ranking with initialized gradient boosted regression trees. In: *Proceedings of the learning to rank challenge*, pp. 77–89 (2011)
21. Page, L., Brin, S., Motwani, R., Winograd, T.: *The Pagerank Citation Ranking: Bringing Order to the Web*. Tech. rep., Stanford InfoLab (1999)
22. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* **12**(Oct), 2825–2830 (2011)
23. Radlinski, F., Joachims, T.: Query chains: learning to rank from implicit feedback. In: *KDD'05*, pp. 239–248. ACM (2005)
24. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-Based collaborative filtering recommendation algorithms. In: *WWW'01*, pp. 285–295 (2001)
25. Sayyadi, H., Getoor, L.: Futurerank: ranking scientific articles by predicting their future pagerank. In: *SDM'09*, pp. 533–544. SIAM (2009)
26. Shang, S., Ding, R., Yuan, B., Xie, K., Zheng, K., Kalnis, P.: User oriented trajectory search for trip recommendation. In: *EDBT*, pp. 156–167 (2012)
27. Shang, S., Ding, R., Zheng, K., Jensen, C.S., Kalnis, P., Zhou, X.: Personalized trajectory matching in spatial networks. *VLDB, J* **23**(3), 449–468 (2014)
28. Surdeanu, M., Ciaramita, M., Zaragoza, H.: Learning to rank answers on large online Qa collections. In: *Proceedings of the meeting of the association for computational linguistics*, pp. 719–727 (2008)
29. Wang, Y., Wang, L., Li, Y., He, D., Chen, W., Liu, T.Y.: A theoretical analysis of Ndcg ranking measures. In: *COLT'13* (2013)
30. Yang, D., Chen, T., Zhang, W., Lu, Q., Yu, Y.: Local implicit feedback mining for music recommendation. In: *Recsys'12*, pp. 91–98. ACM (2012)
31. Yin, P., Luo, P., Lee, W.C., Wang, M.: Silence is also evidence: interpreting dwell time for recommendation from psychological perspective. In: *KDD'13*, pp. 989–997. ACM (2013)
32. Yu, J., Tao, D., Wang, M., Rui, Y.: Learning to rank using user clicks and visual features for image retrieval. *IEEE Transactions on Cybernetics* **45**(4), 767–779 (2015)
33. Zheng, K., Su, H., Zheng, B., Shang, S., Xu, J., Liu, J., Zhou, X.: Interactivses. In: *ICDE*, pp. 423–434 (2015)
34. Zheng, Z., Wu, X., Zhang, Y., Lyu, M.R., Wang, J.: Qos ranking prediction for cloud services. *IEEE Trans. Parallel Distrib. Syst.* **24**(6), 1213–1222 (2013)
35. Zhou, Z.H.: *Ensemble methods: foundations and algorithms*. CRC Press, Boca Raton (2012)