ICIC Express Letters
Part B: Applications
Volume **8**, Number **8**, August **2017**

 ISSN 2185-2766
pp. 1165–1171

LAMBDAXGB: RESEARCH ON LEARNING TO RANK BASED ON LAMBDAMART

Liyan Xiong[1], Xiaoxia Chen[1], Xiaohui Huang[1], Weichun Huang[2]
Maosheng Zhong[1] and Hui Zeng[1]

[1]School of Information Engineering
[2]School of Software Engineering
East China Jiaotong University
No. 808, Shuanggang East Avenue, Nanchang 330013, P. R. China
{ xly_ecjtu; chenxiao970508; hwc1968 }@163.com; hxh016@hotmail.com
zhongmaosheng@sina.com; 331549185@qq.com

Received March 2017; accepted May 2017

Abstract. *In this paper, RankNet, LambdaRank, LambdaMART and the XGBoost are studied and analyzed. The idea of improving the LambdaMART is proposed, that is, adding the regulation to the loss function of LambdaMART. Two commonly regulations L1 and L2 are added to the loss function of the LambdaMART and three algorithms are proposed, including LambdaXGB L1, LambdaXGB L2 and LambdaXGB. Through the MQ2008 dataset, this paper reveals the NDCG evaluation result compared with RankNet and LambdaMART, and verifies the effectiveness of these algorithms. The results demonstrate that our approach gives state-of-the-art results on a rank of dataset.*
Keywords: RankNet, LambdaMART, LambdaXGB, LambdaXGB L1, LambdaXGB L2

1. **Introduction.** With the increasing selection, search engines and recommendation systems are more and more dependent on the sort. However, single factor is only considered by the traditional sorting algorithm. With the exponential growth of processed data, multiple factors need to be combined for sorting, endowed with different weights. So that is something about Learning to Rank [1]. Learning to Rank is a sort of supervised learning method, which can get a rank model according to the training data, and then use this rank model to sort the data.

The pairwise is transformed into binary classification problem in ranking the documents. For the documents of the same query, the training samples of binary classifier training are obtained for any two different labels. All the document pairs are sorted to get a partial order, and the final rank is achieved. The pairwise approach includes RankNet, LambdaRank, LambdaMART, Ranking SVM, IR SVM, RankBoost.

This work contributes to the follow aspect: distinguished from the existing method of LambdaMART, we add the regulation to the loss function of LambdaMART to build the new models, including LambdaXGB L1, LambdaXGB L2 and LambdaXGB. And the experiment demonstrates that our approach gives state-of-the-art results on a rank of problems.

The rest part of the paper is structured as follows. We discuss related work in Section 2. We discuss the LambdaXGB model in Section 3. The experiment and result analysis are shown in Section 4. Finally, we conclude the paper and discuss the directions of the future works in Section 5.

2. **Related Work.** The RankNet is an underlying model, which maps an input feature vector to a number during training. For a given query, inputting the document of the query, there is output ranking model $f(d, w)$. The cross entropy cost function penalizes

the deviation of the model output probabilities from the desired probabilities. The ranking function of RankNet can be learned by neural network. Last, the RankNet model parameters are learning via gradient descent.

LambdaRank, based on RankNet, is a gradient descent method by using an approximation to the NDCG "gradient", and has performed very well in practice [2]. As a general gradient descent optimization framework, LambdaRank needs to define the gradient, not the objective function. In the document pairs $< i, j >$, if i is more relevant than j, the derivative should be adjusted. Then in 2008 Donmez et al. propose LambdaRankNDCG, LambdaRankMAP, and LambdaRankMRR [3].

LambdaMART combines MART (Multiple Additive Regression Trees) [4] and LambdaRank. While LambdaRank was originally instantiated using neural nets, LambdaMART implements the same ideas using the boostedtree style MART algorithm, which itself may be viewed as a gradient descent algorithm [5]. LambdaMART is able to choose splits and leaf values that may decrease the utility for some queries, as long as the overall utility increases [6]. In 2011 Burges et al. used a linear combination of twelve ranking models, eight of which were bagged LambdaMART boosted tree models, two of which were LambdaRank neural nets, and two of which were MART models using a logistic regression cost [7].

XGBoost (eXtreme Gradient Boosting) learns an ensemble of boosted trees which makes careful tradeoff between classification error and model complexity [8]. XGBoost takes a top down approach, by building a scalable tree boosting system on top of a few primitives for which the implementation can be easily replaced [9]. XGBoost, a scalable tree boosting system, is widely used by data scientists and provides state-of-the-art results on many problems [10].

3. The Framework of LambdaXGB.

XGBoost is based on the GBDT model, which is also called CART. So we combine the LambdaMART and XGBoost, and add L1-regularization term and L2-regularization term into the loss function of the LambdaMART. As the result, three algorithms are proposed, including LambdaXGB L1, LambdaXGB L2 and LambdaXGB.

Notation: we denote the cross function by C, the target probabilities of document i are to be ranked higher than document j by P_{ij}, the desired target values by $\overline{P}_{ij}$.

3.1. LambdaXGB L1.

L1-regularization term gives a class of sparse estimates, which contains variable selection to the modeling problem. The L1-regularization term is

$$C = C_0 + \frac{\lambda}{n}|\omega| \tag{1}$$

Since we want to compute C, here define

$$C = \lambda \left| P_{ij} - \overline{P}_{ij} \right| \tag{2}$$

where in the RankNet the learned probability and desired probability are

$$P_{ij} = \frac{1}{1 + e^{-\sigma(s_i - s_j)}} \tag{3}$$

$$\overline{P}_{ij} = \frac{1}{2}(1 + S_{ij}) \tag{4}$$

where $S_{ij} = 1$ in the LambdaRank.

So Equation (2) is replaced by

$$C = \frac{\lambda e^{-\sigma(s_i - s_j)}}{1 + e^{-\sigma(s_i - s_j)}} \tag{5}$$

So we defined the loss function of LambdaXGB L1 as

$$C = \sum_{\{i,j\}\Leftrightarrow I} \left[|\Delta Z_{ij}| \log\left(1 + e^{-\sigma(s_i - s_j)}\right) + \frac{\lambda e^{-\sigma(s_i - s_j)}}{1 + e^{-\sigma(s_i - s_j)}} \right] \tag{6}$$

Taking first derivatives of C with respect to the s_i gives

$$\begin{aligned} \frac{\partial C}{\partial s_i} &= \sum_{\{i,j\}\Leftrightarrow I} \frac{-\sigma\,|\Delta Z_{ij}|}{1 + e^{\sigma(s_i - s_j)}} + \frac{-\sigma\lambda e^{\sigma(s_i - s_j)}}{\left(1 + e^{\sigma(s_i - s_j)}\right)^2} \\ \frac{\partial C}{\partial s_i} &= \sum_{\{i,j\}\Leftrightarrow I} -\sigma\,|\Delta Z_{ij}|\,\rho_{ij} - \sigma\lambda\rho_{ij}^2 \left(\frac{1}{\rho_{ij}} - 1\right) \end{aligned} \tag{7}$$

Also define

$$\rho_{ij} = \frac{1}{1 + e^{\sigma(s_i - s_j)}} \tag{8}$$

Taking second derivatives of C with respect to the s_i gives

$$\frac{\partial^2 C}{\partial s_i^2} = \sum_{\{i,j\}\Leftrightarrow I} \sigma^2\,|\Delta Z_{ij}|\,\rho_{ij}(1 - \rho_{ij}) + \sigma^2\lambda\rho_{ij}^3 \left(\frac{1}{\rho_{ij}} - 1\right)\left(\frac{1}{\rho_{ij}} - 2\right) \tag{9}$$

3.2. **LambdaXGB L2.** L2-regularization term is one of the most common forms of regularization, and is also known as ridge regression. The L2-regularization term is

$$C = C_0 + \frac{\lambda}{2n} \sum \omega^2 \tag{10}$$

In the same way, we define C as

$$C = \frac{1}{2}\lambda \left(P_{ij} - \overline{P}_{ij}\right) \tag{11}$$

So Equation (11) is replaced by

$$C = \frac{1}{2}\lambda \left(\frac{-e^{-\sigma(s_i - s_j)}}{1 + e^{-\sigma(s_i - s_j)}}\right)^2 \tag{12}$$

So we defined the loss function of LambdaXGB L2 as

$$C = \sum_{\{i,j\}\Leftrightarrow I} \left[|\Delta Z_{ij}| \log\left(1 + e^{-\sigma(s_i - s_j)}\right) + \frac{1}{2}\lambda \left(\frac{-e^{-\sigma(s_i - s_j)}}{1 + e^{-\sigma(s_i - s_j)}}\right)^2 \right] \tag{13}$$

Taking first derivatives of C with respect to the s_i gives

$$\begin{aligned} \frac{\partial C}{\partial s_i} &= \sum_{\{i,j\}\Leftrightarrow I} \frac{-\sigma|\Delta Z_{ij}|}{1 + e^{\sigma(s_i - s_j)}} + \frac{-\sigma\lambda e^{\sigma(s_i - s_j)}}{\left(1 + e^{\sigma(s_i - s_j)}\right)^3} \\ \frac{\partial C}{\partial s_i} &= \sum_{\{i,j\}\Leftrightarrow I} -\sigma|\Delta Z_{ij}|\rho_{ij} - \sigma\lambda\rho_{ij}^3 \left(\frac{1}{\rho_{ij}} - 1\right) \end{aligned} \tag{14}$$

Taking second derivatives of C with respect to the s_i gives

$$\frac{\partial^2 C}{\partial s_i^2} = \sum_{\{i,j\}\Leftrightarrow I} \sigma^2|\Delta Z_{ij}|\rho_{ij}(1 - \rho_{ij}) + \sigma^2\lambda\rho_{ij}^4 \left(\frac{1}{\rho_{ij}} - 1\right)\left(\frac{2}{\rho_{ij}} - 3\right) \tag{15}$$

3.3. **LambdaXGB.** LambdaXGB combines LambdaXGB L1 and LambdaXGB L2. So we defined the loss function of LambdaXGB as

$$C = \sum_{\{i,j\}\Leftrightarrow I} \left[|\Delta Z_{ij}| \log\left(1 + e^{-\sigma(s_i - s_j)}\right) + \frac{\lambda e^{-\sigma(s_i - s_j)}}{1 + e^{-\sigma(s_i - s_j)}} + \frac{1}{2}\lambda \left(\frac{-e^{-\sigma(s_i - s_j)}}{1 + e^{-\sigma(s_i - s_j)}} \right)^2 \right] \tag{16}$$

Taking first derivatives of C with respect to the s_i gives

$$\frac{\partial C}{\partial s_i} = \sum_{\{i,j\}\Leftrightarrow I} -\sigma |\Delta Z_{ij}| \rho_{ij} - \sigma\lambda\rho_{ij}^2 \left(\frac{1}{\rho_{ij}} - 1 \right) - \sigma\lambda\rho_{ij}^3 \left(\frac{1}{\rho_{ij}} - 1 \right) \tag{17}$$

Taking second derivatives of C with respect to the s_i gives

$$\begin{aligned} \frac{\partial^2 C}{\partial s_i^2} = \sum_{\{i,j\}\Leftrightarrow I} & \sigma^2 |\Delta Z_{ij}| \rho_{ij}(1 - \rho_{ij}) + \sigma^2\lambda\rho_{ij}^3 \left(\frac{1}{\rho_{ij}} - 1 \right) \left(\frac{1}{\rho_{ij}} - 2 \right) \\ & + \sigma^2\lambda\rho_{ij}^4 \left(\frac{1}{\rho_{ij}} - 1 \right) \left(\frac{2}{\rho_{ij}} - 3 \right) \end{aligned} \tag{18}$$

4. Experimental Results and Analysis.

4.1. **Dataset.** Select the MQ2008 dataset of the LETOR 4.0, including Fold1, Fold2, Fold3, Fold4 and Fold5 five subsets.

4.2. **Evaluations.** The (lack of) ability of a query to rank highly relevant documents toward the top of the result list should show on both the cumulated gain by document rank (CG) and the cumulated gain with discount by document rank (DCG) vectors [11]. DCG is a weighted sum of the degree of relevancy of the ranked items.

NDCG (normalized discounted cumulative gain) has two advantages compared to many other measures. First, NDCG allows each retrieved document has graded relevance while most traditional ranking measures only allow binary relevance. Second, NDCG involves a discount function over the rank while many other measures uniformly weigh all positions [12].

4.3. **Experimental results and analysis.** Ranking accuracy was computed using NDCG@5, NDCG@10, NDCG@15, NDCG@20, NDCG@25 and NDCG@30. The performance of the RankNet, LambdaMART, LambdaXGB L1, LambdaXGB L2 and LambdaXGB on the MQ2008 dataset is as follows.

FIGURE 1. The result of Fold1 train set

According to the figures given in the above chart, on the result of Fold1 train set, the performance of LambdaMART in NDCG@5 is particularly prominent, but on the result of Fold1 validation and test, LambdaMART is worse than the LambdaXGB. The performance of LambdaXGB is better than LambdaMART in NDCG@10, NDCG@15, NDCG@20, NDCG@25 and NDCG@30 on the result of Fold1 train. We conclude that for the Fold1 data set, LambdaXGB achieved the highest score.

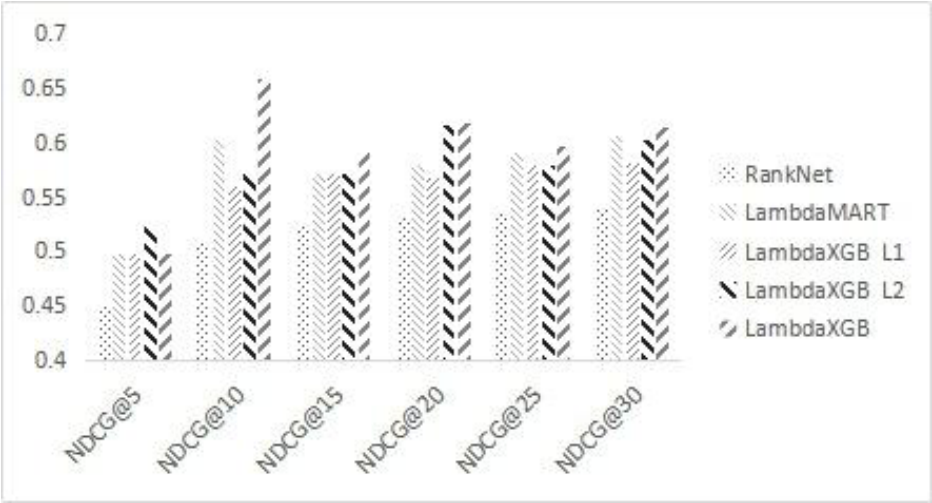

FIGURE 2. The result of Fold2 train set

The above chart shows these methods based on the Fold2 data set, on the result of Fold2 train set, the result of LambdaXGB in NDCG@5 is in agreement with the LambdaMART. The performance of LambdaXGB is better than LambdaMART in NDCG@10, NDCG@15, NDCG@20, NDCG@25 and NDCG@30 on the result of Fold2 train. We conclude that LambdaXGB acquired a great score by beating all other algorithms on the Fold2 data set.

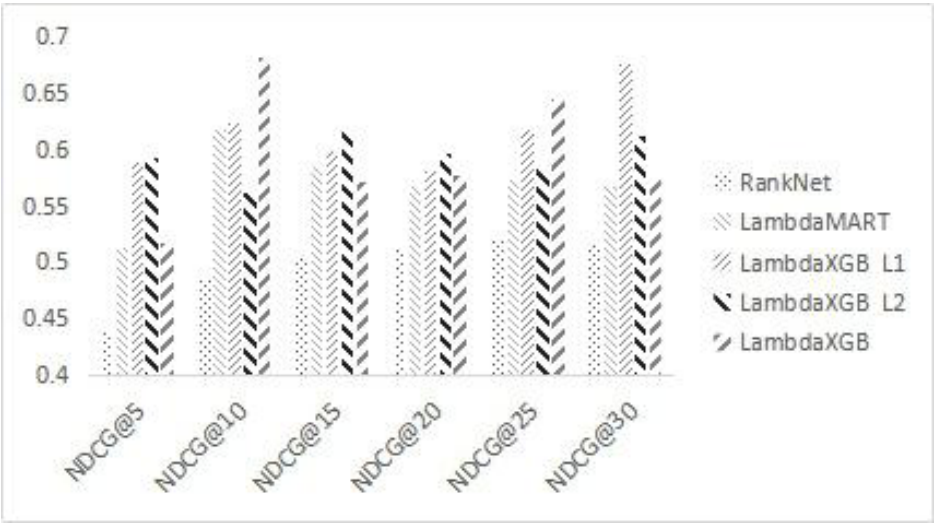

FIGURE 3. The result of Fold3 train set

According to the figures given in the above chart, on the result of Fold3 train set, the result of LambdaXGB in NDCG@5 and NDCG@30 is in agreement with the LambdaMART. LambdaXGB is worse than the LambdaMART in NDCG@15. The performance of LambdaXGB is better than LambdaMART in NDCG@10, NDCG@20 and NDCG@25 on the result of Fold3 train. We conclude that for the Fold3 data set, LambdaXGB achieved the better score than other algorithms.

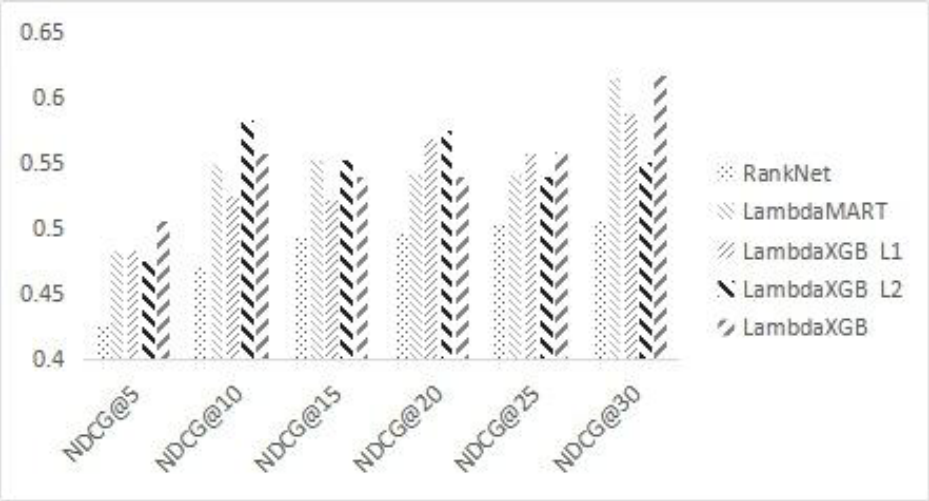

FIGURE 4. The result of Fold4 train set

According to the figures given in the above chart, on the result of Fold4 train set, the result of LambdaXGB in NDCG@20 and NDCG@30 is in agreement with the LambdaMART. LambdaXGB is worse than the LambdaMART in NDCG@15. The performance of LambdaXGB is better than LambdaMART in NDCG@5, NDCG@10 and NDCG@25 on the result of Fold4 train. We conclude that for the Fold4 data set, LambdaXGB achieved the better score than other algorithms.

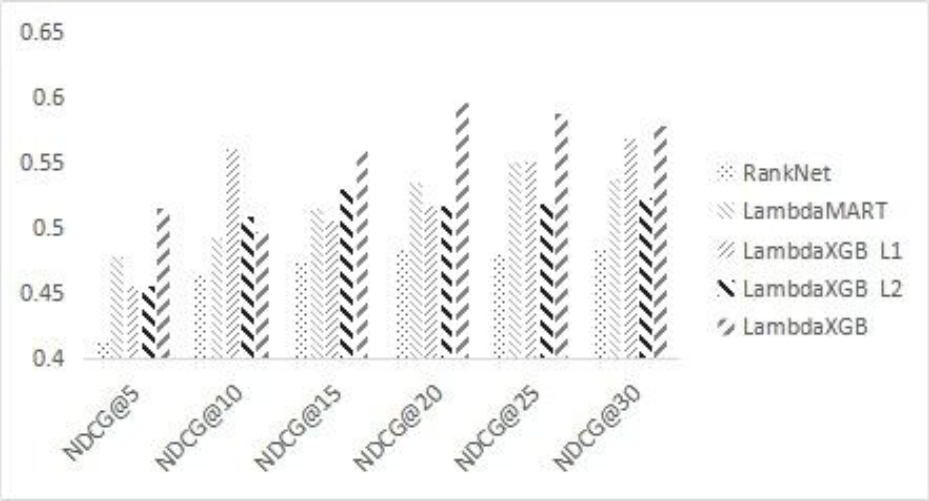

FIGURE 5. The result of Fold5 train set

According to the figures given in the above chart, on the result of Fold5 train set, the result of LambdaXGB in NDCG@10 is in agreement with the LambdaMART. The performance of LambdaXGB is better than LambdaMART in NDCG@5, NDCG@15, NDCG@20, NDCG@25 and NDCG@30 on the result of Fold5 train. We can conclude that for the Fold5 data set, LambdaXGB achieved the better score than other algorithms.

In summary, according to the figures given in the above five charts, on the result of these train sets, LambdaXGB achieved the better score than other algorithms.

5. **Conclusion and Future Works.** In this paper, RankNet, LambdaRank, LambdaMART and the XGBoost are studied and analyzed, and then the idea of improving the LambdaMART is proposed, that is, adding the regulation to the loss function of LambdaMART. There are L1 and L2 of the common regulation, and then the loss function of the LambdaMART adds these two regular terms. Three algorithms are proposed, including

LambdaXGB L1, LambdaXGB L2 and LambdaXGB. Through the MQ2008 dataset, This paper reveals the NDCG evaluation result compared with RankNet and LambdaMART, and verifies these algorithms of our paper. The results demonstrate that our approach gives state-of-the-art results on a rank of problems.

The LambdaXGB gives further improvement. For future work it will be interesting to investigate extending the approach via following three ways. First, the choice of parameters of regulation L1 and L2 makes difference to the LambdaXGB, so we can adjust the parameters to acquire better scores. Second, the results only demonstrated on the MQ2008 data set, and then future work should demonstrate the LambdaXGB on any other data sets. Third, to quickly optimize the objective of the XGBoost, use the second order Taylor expansion. LambdaXGB also can use the second order Taylor expansion to optimize the objective, which is the future work.

Acknowledgement. This research was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61363072, and No. 61562027, the Natural Science Foundation of Jiangxi Province under Grant Nos. 20161BAB212050, 20161BBI90032, Education Department of Jiangxi Province under Grant No. GJJ160508. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] T. Y. Liu, *Learning to Rank for Information Retrieval*, Springer Berlin Heidelberg, 2011.

[2] Y. Yue and C. J. C. Burges, *On Using Simultaneous Perturbation Stochastic Approximation for Learning to Rank, and the Empirical Optimality of LambdaRank*, Microsoft Research Technical Report MST-TR-2007-115, 2007.

[3] P. Donmez, K. M. Svore and C. J. C. Burges, On the local optimality of LambdaRank, *Proc. of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp.460-467, 2009.

[4] J. H. Friedman, Greedy function approximation: A gradient boosting machine, *Annals of Statistics*, vol.29, no.5, pp.1189-1232, 2000.

[5] O. Chapelle and Y. Chang, Yahoo! Learning to rank challenge overview, *Journal of Machine Learning Research*, vol.14, pp.1-24, 2011.

[6] C. J. C. Burges, From RankNet to LambdaRank to LambdaMART: An overview, *Learning*, vol.11, 2010.

[7] C. J. C. Burges, K. M. Svore, P. N. Bennett et al., Learning to rank using an ensemble of lambda-gradient models, *Yahoo! Learning to Rank Challenge*, pp.25-35, 2011.

[8] T. Chen and T. He, Higgs boson discovery with boosted trees, *Proc. of the 2014 International Conference on High-Energy Physics and Machine Learning*, vol.42, pp.69-80, 2014.

[9] T. Chen and C. Guestrin, XGBoost: Reliable large-scale tree boosting system, *Proc. of the 22nd SIGKDD Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, pp.13-17, 2016.

[10] T. Chen and C. Guestrin, XGBoost: A scalable tree boosting system, *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.785-794, 2016.

[11] K. Järvelin and J. Kekäläinen, Cumulated gain-based evaluation of IR techniques, *ACM Trans. Information Systems*, vol.20, no.4, pp.422-446, 2002.

[12] Y. Wang, L. Wang, Y. Li et al., A theoretical analysis of NDCG ranking measures, *Proc. of the 26th Annual Conference on Learning Theory*, 2013.