Comparing Offline and Online Evaluation Results of Recommender Systems

REVEAL Workshop paper

Tomas Rehorek
Czech Technical University, Recombee
tomas.rehorek@recombee.com

Ondrej Biza
Czech Technical University
bizaondr@fit.cvut.cz

Radek Bartyzal
Czech Technical University, Recombee
radek.bartyzal@recombee.com

Pavel Kordik
Czech Technical University, Recombee
kordikp@fit.cvut.cz

Ivan Povalyev
Recombee
ivan.povalyev@recombee.com

Ondrej Podsztavek
Czech Technical University
podszond@fit.cvut.cz

ABSTRACT

Recommender systems are usually trained and evaluated on historical data. Offline evaluation is, however, tricky and offline performance can be an inaccurate predictor of the online performance measured in production due to several reasons. In this paper, we experiment with two offline evaluation strategies and show that even a reasonable and popular strategy can produce results that are not just biased, but also in direct conflict with the true performance obtained in the online evaluation. We investigate offline policy evaluation techniques adapted from reinforcement learning and explain why such techniques fail to produce an unbiased estimate of the online performance in the "watch next" scenario of a large-scale movie recommender system. Finally, we introduce a new evaluation technique based on Jaccard Index and show that it correlates with the online performance.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; • **Theory of computation** → *Reinforcement learning*;

KEYWORDS

Recall, CTR, Recommender Systems, Policy Evaluation

ACM Reference Format:
Tomas Rehorek, Ondrej Biza, Radek Bartyzal, Pavel Kordik, Ivan Povalyev, and Ondrej Podsztavek. 2018. Comparing Offline and Online Evaluation Results of Recommender Systems: REVEAL Workshop paper. In *Proceedings of RecSyS conference (RecSyS'18).* ACM, New York, NY, USA, Article 4, 5 pages. https://doi.org/10.475/123_4

RecSyS'18, , Vancouver, Canada

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

1 INTRODUCTION

Online evaluation is the best approach to assessing the performance of recommender systems, but it poses many challenges: deploying models online is time-consuming, models with poor performance harm user experience and the measurements are irreproducible. Therefore, offline evaluation on historical data is often used to train recommender systems and select candidates that might perform well online. The more offline measures correlate with online results the better.

There are many problems preventing offline evaluation methods from being unbiased estimates of the online performance. Many studies [20] have shown that offline measures such as root-mean-square error (RMSE) on historical ratings are poor estimators. Moreover, it is hard to estimate how the user would have reacted if presented with a different set of recommendations. When an evaluated algorithm generates a different recommendation than the algorithm used to collect historical user interactions, the performance estimate is poor.

Recommendation, similarly as search or any other learning-to-rank problem, can also be viewed as a reinforcement learning problem, where selecting good recommendation leads to higher future rewards (clicks, purchases). Although most companies running recommender systems are oriented towards short-term rewards that are easier to measure (e.g., immediate click-through-rate and conversion rate), optimizing long-term rewards such as customer lifetime value leads to lower churn of users, increased satisfaction and loyalty, and pays off in the long-term. In this paper, we also focus on short-term performance criteria, but there is space for further extension of the proposed method to long-term evaluation.

Recommendation as a reinforcement learning problem has been studied in [2, 23, 28]. It is easy to map a recommendation task into the reinforcement learning domain. Actions are possible recommendations for a given user in a given state, rewards can be derived from implicit user ratings and policies are recommendation algorithms. The main problem is that the number of possible actions (ranked lists of recommended items) can be enormous for real-world recommenders (having millions of items that can be recommended alone, not even taking their combinations into account). Even simple bandit-based reinforcement learning algorithms suffer from scalability issues.

Considering a recommendation task simplified to generate a single item, it is possible to imagine a context-free greedy k-bandit such as the "trending bestseller" model that generates recommendations based on the recent global popularity of items. Such an algorithm is not very competitive in most recommendation scenarios, and hence contextual bandit algorithms [21] should be employed instead. Deep learning embeddings [3] can be utilized to process and represent the context (e.g. a sequence of deep embeddings of purchased items for each user) so the challenge is to predict the reward [24] or the Q-function [28].

Instead of designing a good and scalable reinforcement learning algorithm for recommendation, which is still a work in progress, this paper targets another important challenge: evaluating recommendation algorithms properly on offline data.

Sampling methods use selected historical recommendations to reduce bias. Weighted importance sampling [15] can be viewed as a special case of weighting the error of individual training samples. Doubly robust evaluation [5, 10] is useful when there is either a good model of rewards or a good model of past policy.

In recommender systems, there are different probabilities of the user observing particular an item. When these probabilities can be estimated from historical data, the Inverse-Propensity-Scoring (IPS) estimator [19] can compute an unbiased offline score. However, estimating these probabilities in large-scale dynamic environments is neither practical nor easy. Basic IPS estimators can even have a negative correlation with the online performance as measured in [8].

We experiment with classical content-based and collaborative filtering algorithms for recommendation and run large-scale experiments to show how different offline evaluation strategies correlate with the online performance.

2 RELATED WORK

Said et al. evaluated basic recommendation algorithms from three different open-source frameworks on Movielens and Yelp datasets [17]. They measure how various aspects of evaluation, including strategies for data splitting (e.g. cross-validation vs. 80%-20% split) and candidate item generation, affect prediction accuracy (RMSE), ranking quality (nDCG@10), catalog coverage and running times. Their unified evaluation uncovered significant differences in prediction accuracy between different implementations of the same algorithms.

Offline evaluation of Contextual Bandits was studied in [5–7, 9, 11–14, 16, 27]. A replay-based evaluation method was first proposed in [11] and further studied in [12, 13]. The method considers only the logged data that match the recommendations of the evaluated model. [13] proved their evaluator is unbiased given an infinite data stream of i.i.d. events from a uniformly random logging policy. A common trait of replay-based evaluators is that only a fraction of events generate the final score. This can cause the evaluator to be biased towards short sequences of events (because the data stream in never infinite), as discussed in [16], where controlling the bias with bootstrapping techniques is suggested.

A benefit of replay-based methods over simulating the environment is that we can avoid modeling bias. [5] combined a model of the reward function with Importance Sampling to form a Doubly Robust estimator that mitigates the bias introduced by the model and the high variance of Importance Sampling. [10] derived the Doubly Robust estimator for the full reinforcement learning problem and [7, 25, 27] proposed further improvements to the estimator. A lower bound on a return of a trajectory (a sequence of recommendations for a single user) based on Importance Sampling was derived in [26] and compared with online evaluation in [24].

Handling selection bias in the evaluation and training of recommender systems was explored in [19]. The introduced approach is based on Propensity Scoring, where propensities were estimated by Naive Bayes. Results on two datasets indicate that bias is reduced, but the online performance was not measured to confirm the hypothesis.

In [22] offline evaluation for slates recommendation is discussed. The number of possible slates (ranked lists of recommended items) is almost infinite given the number of items in real-world databases. It is also not practical to assume that we can estimate the probability that a particular state is generated given complex recommendation algorithms and a high number of slates.

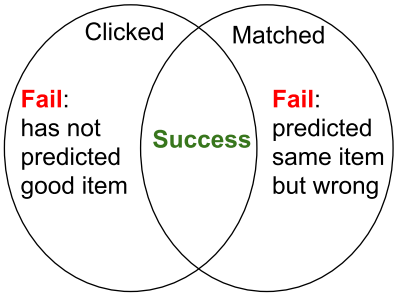

Figure 1: The Jaccard index maximizes the relative size of the region where new algorithm matches the old one on the first recommendation when it was successful (the user has clicked). The other regions are considered a failure. Recommendations that are both unmatched and unclicked are not taken into account, because there is no hint if they can succeed.

3 OUR APPROACH

We argue that all the above-mentioned approaches either do not give us an unbiased estimate of the online performance or come with too strong assumptions that are inappropriate or hardly applicable in production environments. When offline data are generated by standard collaborative filtering based algorithms, most of the assumptions that were used to derive the estimators are violated.

We designed and explored several estimators and found out that one performs particularly well. We extended an IPS estimator (Algorithm 2 from [12]) which is penalizing algorithms similar to the one used to obtain offline data as we explain in the next Section.

Our Jaccard Index based Estimator (JIE) reduces this penalty by normalizing successful hits (number of recommendations with matched first item followed by a click or a conversion) by unsuccessful attempts when the evaluated algorithm a) has not recommended clicked first item successful predicted by online data producer, or b) recommended the same item as online data producer but there was no click or conversion (see Figure 1).

$$JIE = \frac{\text{clicked} \cap \text{matched}}{\text{clicked} \cup \text{matched}}$$

Compared to [13], we do not assume i.i.d. generation process. This has two important consequences. 1) The online performance estimation is biased towards the online generation process, such as the currently deployed collaborative filtering algorithm. Specifically, we are possibly unfairly penalizing models that would have good performance, yet by recommending completely different items. This disadvantage is, however, compensated by 2) There is no need to expose users to random recommendations, significantly damaging their trust in the recommendations and possibly the product image of the whole system. In scenarios like similar/related items recommendation, using a random model is hardly possible.

To compute JIE, we iterate through all recommendations generated by the original model $model_o$ during a selected, recent time period. We denote R_{orig} the set of records containing collected information about these recommendations. Each entry $(user_o, time_o, recom_o, clicked_o) \in R_{orig}$ holds information that $user_o$ has been shown $recom_o$ as the first recommend item at $time_o$ with boolean flag $clicked_o$ determining whether the user was a reward or not. For each record in R_{orig}, we generate alternative recommendation $recom_i$ by all the models from M, simulating exactly the same conditions to those that were present when generating $recom_o$ by production model at $time_o$. This includes hiding all interaction data that appeared after $time_o$ and using the exact same business rules applied to the corresponding recommendation request.

Finally, we compute JIE for all the alternative models $model_i \in M$ by aggregating numbers from cases when there is either match between $recom_o$ and $recom_i$, or when $recom_o$ has been clicked, considering successful only the cases when both match and click happened. See the JIE computation methodology in Alg. 1 below.

Algorithm 1 Jaccard Index based Estimator computation

for $model_i \in M$ **do**
  $success_i \leftarrow 0$
  $clicked_i \leftarrow 0$
end for
for $(user_o, time_o, recom_o, clicked_o) \in R_{orig}$ **do**
  for $model_i \in M$ **do**
    $recom_i \leftarrow model_i(user_o, time_o)$
    $matched_i \leftarrow recom_o = recom_i$
    if $clicked_o \vee matched_i$ **then**
      $total_i \leftarrow total_i + 1$
      if $clicked_o \wedge matched_i$ **then**
        $success_i \leftarrow success_i + 1$
      end if
    end if
  end for
end for
return $\left(\frac{success_i}{total_i}, \ldots, \frac{success_{|M|}}{total_{|M|}}\right)$

4 EXPERIMENTS

An important contribution of our work is that we were able to validate our theoretical hypotheses in a large-scale production environment. We are aware of the limited reproducibility of our results; however, it is hard to reproduce online tests with real users. We believe that our findings are still interesting for the research community and can be reproduced by another team with access to a large-scale recommendation infrastructure.

Our aim was to measure the correlation between the proposed Jaccard Index based Estimator (JIE) and the true online performance (CTR) of candidate models. Our client Showmax agreed with online experiments on a small portion of the "watch next" recommendation scenario to verify the hypotheses discussed in this paper. Henceforth, during the evaluation period, we let the current production model ($model_o$) generate the recommendations for the majority of users, but for a limited subset of users, we deployed individual candidate models $model_i \in M$. These models were evaluated both online (measuring true CTR) and offline by JIE. Thanks to this, we are able to compare the estimation performance of JIE.

One of the challenges is that our customers use a query language (ReQL) on top of each recommendation request, allowing them to filter out or boost particular items and more. It is crucial to exactly emulate ReQL business rules offline to get results comparable to the online behavior of algorithms under investigation. This again complicates reproducibility and generalization of our results to other recommendation scenarios with different dynamics. Nevertheless, we believe our results are valuable with reasonable chance that the hypotheses holding for a particular "watch next" scenario will hold for other scenarios as well.

The model currently used in production is an improved version of Collaborative Filtering User-kNN algorithm with cosine similarity and Non-normalized Cosine Neighborhood as defined in [4], using aggregated implicit ratings. The modification is based on using attribute-based or popularity-based models when there is not enough confidence. But considering the given scenario, data density, and used ReQL business rules, the difference between pure and our modified version of User-kNN is small in the most cases.

The models in M we decided to evaluate were:

- *user-knn* – a pure form of Collaborative Filtering User-kNN algorithm with cosine similarity and Non-normalized Cosine Neighborhood as defined in [4], nearly identical to the one running in production,
- *rating-itemknn* – Item-Based Collaborative Filtering k-Nearest Neighbor with cosine similarity as defined in [18],
- *token-itemknn* – Item-Based k-Nearest Neighbor algorithm as defined in [18], but with significantly different similarity measure $sim(i, j)$ working with item attributes, making the algorithm Content-Based rather then Collaborative Filtering. Specifically, Showmax has a mixture of categoric attributes, tags, and text descriptions, all of which are parsed and converted to a common set of *tokens* for each item. When measuring the similarity between two items i and j, modified Jaccard similarity with a TF-IDF-based weighting of individual tokens is used.

We compare online and offline evaluation results for the three models using two different offline evaluation algorithms. We chose Algorithm 2 from [13] as a baseline. The algorithm assumes the logging policy is uniformly random, which is not the case in our experiments. To correct for the bias introduced by the production

Table 1: Online test results on first recommended item

model name	total recomms	actions	CTR (%)
rating-itemknn	9580	368	3.84
token-itemknn	9829	537	5.46
user-knn	9476	588	6.21
default	168848	10234	6.06

Table 2: Bias in offline results - user-knn similar to default recommendations leading to much more matches but also less percentage of actions

model name	total recomms	actions	CTR (%)
rating-itemknn	9536	352	3.69
token-itemknn	12937	668	5.16
user-knn	109027	1372	1.26

Table 3: Jaccard Index Estimator correlates with the online performance

algorithm	recomms	hits	$match \cup click$	JIE
rating-itemknn	148337	372	16681	0.022
token-itemknn	148340	252	11104	0.022
user-knn	148327	2228	76698	0.029

model, we use the Jaccard Index based Estimator described above as the second evaluation algorithm.

5 RESULTS

The results of the online test match our expectations (Table 1). The performance of the user-knn was not statistically different from the default recommendation policy. Token-based itemknn performed slightly worse and the worst performer was the rating-itemknn.

Table 2 shows offline estimates measured using the Algorithm 2 in [13]. The results are strongly biased, greatly underestimating the performance of the user-knn. The source of the bias is the policy that generated the offline data: instead of a randomly uniform logging policy, the recommendations were generated by an ensemble of the user-knn and other methods. As you can see, the number of matching recommendations for the user-knn is much higher than for the other two algorithms. The problem is that, for rating- and token-itemknn, we only consider recommendations that match the logging policy and in such cases (when two diverse algorithms compromise on the first recommendation) the confidence of the recommendation is high, hence higher CTR and biased estimate.

Table 3 shows offline estimates by our proposed JIE method implemented by Algorithm 1. In the $match \cup click$ column showing denominators of Jaccard similarity, we can see that $token - itemknn$ has much higher overlap with the production algorithm than $rating - itemknn$ on the first recommended item. The biggest with $user - knn$ is orderly larger because the two algorithms are nearly identical.

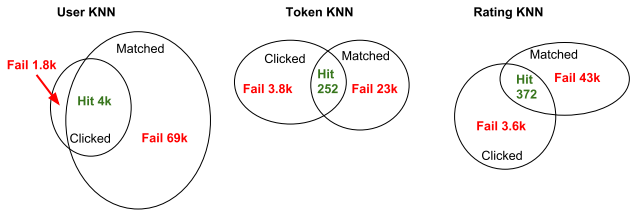

Figure 2: Whereas $userknn$ matches most of the recommendations produced by default policy, the overlap in clicks is also high. For $token - itemknn$ as a different algorithm, the number of matching first items was significantly smaller and a lot of clicks was not predicted. Even worse match and fails in prediction was measured for the $rating - itemknn$.

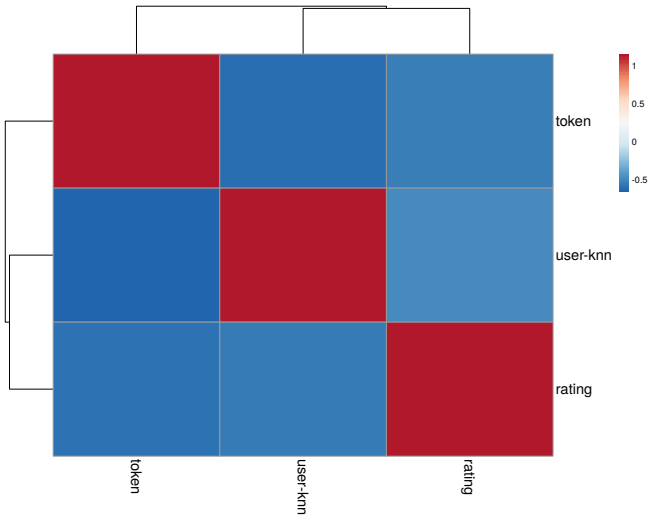

Figure 3: Heatmap of matched recommendation between logging and evaluated policy shows that token and user item-knn share slightly more recommendations than user and rating knn. Proportions of matched recommendations can be used to evaluate diversity of policies.

Similar results can be observed in Figure 2 decomposing JIE to components for an independent evaluation run. Again, offline results correlate with the online performance of the algorithms.

Finally, we decided to run an additional experiment on different recommendation scenario. Contrary to the watch next scenario, the selected scenario presented many items in a row so the visual dominance of first recommended items was absent. We found out, that for this scenario, it is beneficial to compute JIE not just from the single first item, but from K first items displayed to users.

In this scenario, we count match as number of corresponding items between the online and evaluated policy for each recommendation. For match equal to 1, recommendations have to be identical. Zero match means no overlap. We summarized and normalized matches for all recommendations and for all policies in a cross-validation manner.

Figure 3 shows that all three policies are significantly different. The strongest match was always when policies were identical. The number of matched items is also symmetrical following theoretical expectations.

6 DISCUSSION

The proposed offline evaluation methodology and the Jaccard Index Estimator is not completely unbiased.

One possible problem arises when the recommendation algorithm is largely different from the logging policy. The number of matches will be very low in this case and so the number of hits and the confidence of our estimator. We will perform additional experiments to explore such cases.

7 CONCLUSION

Estimation of the online performance from offline data is a difficult task. The main contribution of this paper is that we measured and explained the bias of existing evaluation methods. We showed that the best correlation with the online performance was achieved by Jaccard Index between successful conversions and corresponding recommendations of the evaluated algorithm and the algorithm used to obtain the offline data. Our future work is to investigate the proposed estimator in a much broader experimental setup with hundreds of policies and tens of different recommendation scenarios. We also plan to study how to further improve the robustness of estimates by incorporating propensity scoring where propensities will be estimated by a recently proposed approach [1].

8 ACKNOWLEDGEMENT

This research was partially supported by grant no. GA201/05/0325 of the Grant Agency of the Czech Republic.

REFERENCES

[1] Aman Agarwal, Ivan Zaitsev, and Thorsten Joachims. 2018. Consistent Position Bias Estimation without Online Interventions for Learning-to-Rank. *arXiv preprint arXiv:1806.03555* (2018).

[2] Sungwoon Choi, Heonseok Ha, Uiwon Hwang, Chanju Kim, Jung-Woo Ha, and Sungroh Yoon. 2018. Reinforcement Learning based Recommender System using Biclustering Technique. *arXiv preprint arXiv:1801.05532* (2018).

[3] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. New York, NY, USA.

[4] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of Recommender Algorithms on Top-n Recommendation Tasks. In *Proceedings of the Fourth ACM Conference on Recommender Systems (RecSys '10)*. ACM, New York, NY, USA, 39–46. https://doi.org/10.1145/1864708.1864721

[5] Miroslav Dudík, John Langford, and Lihong Li. 2011. Doubly Robust Policy Evaluation and Learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning (ICML'11)*. Omnipress, USA, 1097–1104. http://dl.acm.org/citation.cfm?id=3104482.3104620

[6] Miroslav DudÃŋk, Dumitru Erhan, John Langford, and Lihong Li. 2014. Doubly Robust Policy Evaluation and Optimization. *Statist. Sci.* 29, 4 (11 2014), 485–511. https://doi.org/10.1214/14-STS500

[7] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. 2018. More Robust Doubly Robust Off-policy Evaluation. *CoRR* abs/1802.03493 (2018). arXiv:1802.03493 http://arxiv.org/abs/1802.03493

[8] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline A/B testing for Recommender Systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 198–206.

[9] William Hoiles and Mihaela Van Der Schaar. 2016. Bounded Off-policy Evaluation with Missing Data for Course Recommendation and Curriculum Design. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. JMLR.org, 1596–1604. http://dl.acm.org/citation.cfm?id=3045390.3045559

[10] Nan Jiang and Lihong Li. 2016. Doubly Robust Off-policy Value Evaluation for Reinforcement Learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. JMLR.org, 652–661. http://dl.acm.org/citation.cfm?id=3045390.3045460

[11] John Langford, Alexander Strehl, and Jennifer Wortman. 2008. Exploration Scavenging. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. ACM, New York, NY, USA, 528–535. https://doi.org/10.1145/1390156.1390223

[12] Lihong Li, Wei Chu, John Langford, Taesup Moon, and Xuanhui Wang. 2012. An unbiased offline evaluation of contextual bandit algorithms with generalized linear models. In *Proceedings of the Workshop on On-line Trading of Exploration and Exploitation 2*. 19–36.

[13] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. 2011. Unbiased Offline Evaluation of Contextual-bandit-based News Article Recommendation Algorithms. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining (WSDM '11)*. ACM, New York, NY, USA, 297–306. https://doi.org/10.1145/1935826.1935878

[14] Lihong Li, Remi Munos, and Csaba Szepesvari. 2015. Toward Minimax Off-policy Value Estimation, In Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS). https://www.microsoft.com/en-us/research/publication/toward-minimax-off-policy-value-estimation/

[15] A Rupam Mahmood, Hado P van Hasselt, and Richard S Sutton. 2014. Weighted importance sampling for off-policy learning with linear function approximation. In *Advances in Neural Information Processing Systems*. 3014–3022.

[16] Olivier Nicol, Jérémie Mary, and Philippe Preux. 2014. Improving offline evaluation of contextual bandit algorithms via bootstrapping techniques. In *International Conference on Machine Learning (Journal of Machine Learning Research, Workshop and Conference Proceedings; Proceedings of The 31st International Conference on Machine Learning)*, Eric Xing and Tony Jebara (Eds.), Vol. 32. Beijing, China. https://hal.inria.fr/hal-00990840

[17] Alan Said and Alejandro Bellogín. 2014. Comparative recommender system evaluation: benchmarking recommendation frameworks. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 129–136.

[18] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW '01)*. ACM, New York, NY, USA, 285–295. https://doi.org/10.1145/371920.372071

[19] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations As Treatments: Debiasing Learning and Evaluation. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. JMLR.org, 1670–1679. http://dl.acm.org/citation.cfm?id=3045390.3045567

[20] Harald Steck. 2011. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*. ACM, 125–132.

[21] Richard S. Sutton and Andrew G. Barto. 1998. Reinforcement Learning: An Introduction. *IEEE Transactions on Neural Networks* 16 (1998), 285–286.

[22] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miro Dudik, John Langford, Damien Jose, and Imed Zitouni. 2017. Off-policy evaluation for slate recommendation. In *Advances in Neural Information Processing Systems*. 3632–3642.

[23] Nima Taghipour, Ahmad Kardan, and Saeed Shiry Ghidary. 2007. Usage-based web recommendations: a reinforcement learning approach. In *Proceedings of the 2007 ACM conference on Recommender systems*. ACM, 113–120.

[24] Georgios Theocharous, Philip S Thomas, and Mohammad Ghavamzadeh. 2015. Personalized Ad Recommendation Systems for Life-Time Value Optimization with Guarantees.. In *IJCAI*. 1806–1812.

[25] Philip Thomas and Emma Brunskill. 2016. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*. 2139–2148.

[26] Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. 2015. High-Confidence Off-Policy Evaluation. https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/10042

[27] Yu-Xiang Wang, Alekh Agarwal, and Miro DudÃŋk. 2017. Optimal and Adaptive Off-policy Evaluation in Contextual Bandits, In Proceedings of the 34th International Conference on Machine Learning. 70, 3589–3597. https://www.microsoft.com/en-us/research/publication/optimal-adaptive-off-policy-evaluation-contextual-bandits/

[28] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A Deep Reinforcement Learning Framework for News Recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 167–176.