Monte Carlo Estimates of Evaluation Metric Error and Bias

Work in Progress

Mucun Tian
People & Information Research Team
Boise State University
Boise, ID, USA
mucuntian@u.boisestate.edu

Michael D. Ekstrand
People & Information Research Team
Boise State University
Boise, ID, USA
michaelekstrand@boisestate.edu

ABSTRACT

Traditional offline evaluations of recommender systems apply metrics from machine learning and information retrieval in settings where their underlying assumptions no longer hold. This results in significant error and bias in measures of top-N recommendation performance, such as precision, recall, and nDCG. Several of the specific causes of these errors, including popularity bias and misclassified decoy items, are well-explored in the existing literature. In this paper we survey a range of work on identifying and addressing these problems, and report on our work in progress to simulate the recommender data generation and evaluation processes to quantify the extent of evaluation metric errors and assess their sensitivity to various assumptions.

KEYWORDS

simulation

1 INTRODUCTION

Traditional offline experiments to evaluate top-N performance of recommender systems typically use metrics and methodologies borrowed from information retrieval and machine learning. The evaluation procedure partitions the user consumption data (such as movie ratings, music likes, etc.) into training and test sets, trains recommendation algorithms on the training set, generates recommendation lists from a set of candidate items for each user, and tests the retrieval or ranking accuracy using the withheld test data as ground truth.

Recommendation scenarios rarely have complete ground truth data, however. This is particularly true for data sets commonly used for offline evaluation in academic research. The standard procedure is to assume that items the user has never rated or consumed are irrelevant; while this assumption is true most of the time, it fails to hold in critical situations that severely undermine the external validity of classical offline evaluations.

In this paper, we present our approach to quantifying the extent and impact of these errors. This ongoing work will yield better insight into precisely how erroneous current evaluation practices are, and we hope that it will also yield statistical techniques and experimental designs to compensate for these errors.

REVEAL'18, October 7, 2018, Vancouver, BC, Canada

This work complements other lines of research that tackle the validity problems in offline evaluation with less-biased estimators [14], protocol tweaks [3, 7, 8], and reframing the problem [4, 12].

2 PROBLEM FRAMING

To measure the accuracy of a recommendation list with a metric such as precision, nDCG, or MRR, the evaluation procedure requires relevance data for all recommended items: for each item, it needs to know whether or not the item is relevant to the user and/or context, or its degree of relevance for metrics that can use such information. This information (or a reasonable proxy for it) is available in certain settings, such as supervised machine learning and TREC competitions. Recommendation scenarios, however, lack such data, and information retrieval techniques for obtaining it such as pooling and relevance imputation are not usually appropriate for recommendation scenarios due to the personalized nature of relevance. This leaves us with two primary solutions:

(1) If known relevance judgments are relative (e.g. ratings), restrict the recommender to rank only items with known relevance, and compare its results against the relevance judgments (*rank effectiveness*).
(2) Assume unrated or unconsumed items are irrelevant.

Most evaluations make assumption (2). This assumption is generally true for individual items [8]; however, it is incorrect in key ways that undermine the validity of resulting evaluations.

Misclassified decoys arise when the user would like an item, but has never rated it in either the training or the test item. This item should be considered a good recommendation, but the evaluation will treat it as irrelevant and penalize the recommender for recommending it. In applications where the recommender should help the user discover new items, this is deeply problematic: a recommender that can accurately find those items the user would like, but may never have found through their existing discovery channels, will perform worse than one that can replicate the user's existing knowledge.

Popularity bias is the effect that evaluations favor algorithms that recommend popular items significantly beyond the intrinsic usefulness of popularity as a recommendation signal. It arises because popular items are more likely to be rated or consumed in both the training and test data; a popular recommendation is therefore a correct answer more often than an unpopular one just because it is popular, not because it matches user taste.

Combined, these problems cause significant challenges for evaluating recommender effectiveness. Popularity bias rigs the evaluation in favor of certain recommendation signals, and misclassified decoys keep the evaluation from identifying and rewarding

an algorithm that can substantially outperform existing social or algorithmic discovery mechanisms.

The exact impact of these problems on metric errors remains unknown, and we seek to estimate it.

3 RELATED WORK

There are several existing approaches to addressing or measuring these problems that inform and complement our work.

3.1 Changing the Protocol

One proposed solution to evaluation difficulties is to change the protocol, particularly the way that test items and candidate items are selected, to neutralize the problem. [1] proposed alternative data split strategies to address popularity bias, both focused on compensating for the rate at which different items appear in the test set. One is to aggregate evaluation metrics by popularity quantile; this enables analysis of the algortihm's effectiveness at different popularity levels, and ensures that the least popular quantile of items influences the final score as much as the most popular quantile. The second is to sample the test data so that each item appears as a test item an equal number of times; grouping data by item and sampling N users who have rated that item will accomplish this. These methods affect absolute metric values, but not necessarily the relative performance of algorithms [2].

Similarly, changing how candidate items — the items the recommender considers when producing its top-N list — are selected may be useful in addressing misclassified decoys. Typically, evaluations use all items that aren't in the user's set of training ratings as candidates; using a candidate set consisting of the user's test items plus a random sample of unrated items decreases the likelihood of misclassified decoys, because if unknown relevant items are relatively rare, they are probably not going to be picked as a part of the sample [7]. However, this method's usefulness relies on unrealistically strong assumptions of the rareness of unknown relevant items, and it likely exacerbates popularity bias [8].

3.2 Unbiased Estimators

Another proposed solution is to select evaluation metrics that admit statistically unbiased estimators using the observed data. Under assumptions that (1) ratings for relevant items are missing at random and (2) the non-relevant ratings have a higher probability of being missing than the relevant ones, computing top-k hit rate (recall) using observed data is an unbiased estimator for the true value [14]. Under the same assumptions, computing non-normalized discounted cumulative gain with observed implicit feedback data is an unbiased estimator for the true value based on complete data [13].

There are two significant limitations to this approach. First, it limits the choice of metrics; in assessing recall, Steck [14] observes that computing precision with observed data is not an unbiased estimator. Lim et al. [13] show that the more common normalized discounted cumulative gain is biased, so that producing an unbiased estimate requires sacrificing normalization. In general, therefore, this approach requires a tradeoff between statistical validity and appropriateness of the metric to the task. If the recommendation task is best captured by a metric without an unbiased estimator, then effectiveness for that task cannot be reliably assessed.

Second, the necessary assumptions are unlikely to hold in realistic scenarios. Ratings or consumption events are not sampled at random from the relevant items; the user's choice of items to rate is based on a complex discovery process based on user knowledge, social networks, and existing discovery tools.

3.3 Counterfactual Learning and Evaluation

One particularly powerful means of addressing the weaknesses of offline evaluation is to reframe the recommendation and evaluation problem as a counterfactual learning problem [4, 9, 15]. This approach aims to reconstruct from offline data an estimate of how the user would have responded had they received a different recommendation. Counterfactual evaluation has the enormous benefit of actually measuring the problem that we most often care about, particularly from business and user response perspectives: the ability to recommend items the user will accept.

Its downside is that it represents a substantial break from historical practice and often is not applicable to commonly-used data sets. While we should — and do — welcome such breaks when they move the field forward substantially, we would also like to understand how much knowledge under the old paradigm can be carried forward, and develop techniques when possible that can be used with more common data sets. The largest available data set for contextual evaluation, from Criteo [9], is valuable but also opaque: the lack of descriptors for item features means that less insight can be obtained about algorithm behavior and performance.

4 SIMULATING EVALUATION

Neither reframing the problem to avoid the pitfalls of classical evaluation nor choosing demonstrably unbiased estimators answers a key question for interpreting previous results: just how wrong are they? Further, the widespread availability of data, metrics, instructions, and tools for classical evaluations makes them relatively easy to perform; if there is a way to improve their accuracy that can be deployed in existing scenarios, such techniques would significantly improve the reliability of recommender systems research and testing.

The most promising technique we see for this work is simulation. Since, by its very nature, we cannot know the underlying ground truth for observed data, and we do not know the particular process by which the observed data was generated, we can't (except in a few limited circumstances) look behind the data to compare observed metric values to what they would be if we had complete relevance data. Simulation, however, lets us open the curtain: by generating complete and observed data under a range of scenarios, we can look at how the observed results vary based on different possible observation processes.

4.1 Existing Simulations

Cañamares and Castells [5] built a probabilistic model to analyze the conditions that determine the usefulness of popularity in recommender systems and better understand popularity bias under various conditions. They defined optimal ranking strategies that maximize the true or observed precision for non-personalized recommendation. By changing the conditional independence among three variables —- item relevance, item discovery, and item rating

— the authors analyzed how the popular recommender and the average rating recommender perform compared to optimal and random recommenders under both observable and true precision. They found that the most-popular recommender is close to the optimal recommender in observed precision and the average-rating recommender is close to optimal in true precision if rating presence is conditionally independent of relevance or no independence assumptions are made. Their analysis implicitly assumes that each of those three variables for every user is independent and identically distributed. They also created a complete data set by asking users to rate songs and indicate whether they had heard the song before rating, allowing for empirical confirmation of the theoretical results. With this data, they found concurring results: the popular recommender works better than the average rating recommender and both recommenders outperform the random recommender in precision and nDCG computed on observable data (only considering relevance on the ratings that the user have heard before). But when using the complete data, the popular recommender is worse than the average rating recommender, and its precision is even worse than that of the random recommender. They also found that using complete data instead of observable data changes the relative performance of collaborative filtering algorithms in some cases.

4.2 Simulation Goals

Building on Cañamares and Castells [5], we are using simulations to address several questions about the error in evaluation metrics (the difference between their values under observable data and their values under complete data):

(1) How is metric error distributed for commonly-used evaluation metrics?
(2) How does metric error distribution change as we change the underlying relevance distribution (data generation process)?
(3) Is the metric error distribution stable across recommendation algorithms?
(4) What ranges of data generation process structures and parameters produce observable data sets comparable to existing data sets on key statistics such as item popularity distribution?
(5) What effect do assumptions such as the independence assumptions in Cañamares and Castells' work have on error distributions?

We hope to use this knowledge to adjust evaluation and analysis techniques to compensate for the observed effects, but documentation of the extent of the problem is a useful research outcome even if it does not seem to be solvable.

4.3 Simulation Approach

We address these questions by simulating the recommender evaluation process, from data generation to metric computation, under controlled conditions. This will allow us to estimate the effect of variations in different stages of an offline evaluation on its accuracy.

When finished, our simulation code will enable us to configure the following:

(1) Underlying true user-item relevance distribution, including models with correlated preferences.
(2) Data observation process, resulting in data comparable to a typical recommender evaluation data set.
(3) Experimental data splitting strategy.
(4) Recommendation technique, including oracle recommenders that have access to the true data.
(5) Evaluation metrics, using both true and observable data.

Step (2) provides a calibration point, as we can compare key statistics such as item popularity and co-rating distributions between the output of our data simulation process and existing data sets such as MovieLens, Last.fm, and the Amazon Reviews data. Producing multiple simulated data sets comparable to published data, but differing in their underlying relevance distributions and observation processes, will enable us to quantify the extent to which violations of experiments' assumptions about the data generation process invalidate their results.

Step (4) will allow us to address important questions such as how often an experimental protocol will reject a perfect recommender; probabilistic oracle recommenders will enable us to answer questions such as 'if a recommender has precision of 80%, with errors randomly distributed, what will its observed precision be?'.

5 DATA AND TOOLS

We have several common data sets on hand for calibrating and tuning our observed data simulators, including:

(1) MovieLens [10]
(2) BookCrossing [16]
(3) Amazon Reviews [11]
(4) Last.fm music play counts [6]
(5) ACM Digital Library metadata, including the citation graph for older articles

We will also use the data collected by Cañamares and Castells [5] for further calibration and confirmation of our results.

We are implementing our simulation code in Python, and will make this code available upon publication of results. We will leverage the LKPY recommender toolkit for evaluation metrics and collaborative filtering implementations.

6 EARLY RESULTS

We have begun work on simulating users' true preferences by uniform selection and an Indian buffet process, and sampling observations from the true preferences by uniform sampling and popularity-weighted sampling. We then use an oracle recommender to produce optimal recommendations and compute common evaluation metrics including precision, recall, and nDCG. with both observed and complete data. Consistent with the findings of [14], we find observed recall values generally symmetrically distributed around true recall. Precision and nDCG show large deviations between metric distributions computed from observed data and complete data.

7 CONCLUSION

This work will improve the state of the art in offline evaluation by shedding light on just how broken existing practice is. We know that it has significant conceptual problems, but do not yet have extensive data on the statistical impact of those problems.

Understanding these problems will help us better interpret existing research findings and hopefully enable us to adapt experimental designs to compensate for evaluation biases. It will complement other important work such as counterfactual evaluation by building a bridge between historical research practices and more sophisticated understandings of recommendation problems and evaluation techniques.

This work is ongoing, and we invite feedback to shape it as we move forward.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. IIS 17-51278.

REFERENCES

[1] Alejandro Bellogín. 2012. Recommender System Performance Evaluation and Prediction: An Information Retrieval Perspective. http://ir.ii.uam.es/~alejandro/thesis/thesis-bellogin.pdf

[2] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2011. Precision-oriented Evaluation of Recommender Systems: An Algorithmic Comparison. In *Proceedings of the Fifth ACM Conference on Recommender Systems (RecSys '11)*. ACM, New York, NY, USA, 333–336. https://doi.org/10.1145/2043932.2043996

[3] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2017. Statistical biases in Information Retrieval metrics for recommender systems. *Information Retrieval Journal* (July 2017), 1–29. https://doi.org/10.1007/s10791-017-9312-z

[4] Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X. Charles, D. Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. 2013. Counterfactual reasoning and learning systems: The example of computational advertising. *Journal of machine learning research: JMLR* 14, 1 (2013), 3207–3260. http://www.jmlr.org/papers/volume14/bottou13a/bottou13a.pdf

[5] Rocío Cañamares and Pablo Castells. 2018. Should I Follow the Crowd?: A Probabilistic Analysis of the Effectiveness of Popularity in Recommender Systems. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. ACM, New York, NY, USA, 415–424. https://doi.org/10.1145/3209978.3210014

[6] Òscar Celma. 2010. *Music Recommendation and Discovery in the Long Tail*. Springer.

[7] 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

[8] Michael D. Ekstrand and Vaibhav Mahant. 2017. Sturgeon and the Cool Kids: Problems with Random Decoys for Top-N Recommender Evaluation. In *Proceedings of the Thirtieth International Florida Artificial Intelligence Research Society Conference*. Association for the Advancement of Artificial Intelligence, 639–644. https://aaai.org/ocs/index.php/FLAIRS/FLAIRS17/paper/view/15534

[9] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline A/B testing for Recommender Systems. (Jan. 2018). arXiv:stat.ML/1801.07030 http://arxiv.org/abs/1801.07030

[10] F Maxwell Harper and Joseph A Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems* 5, 4 (Dec. 2015), 19:1–19:19. https://doi.org/10.1145/2827872

[11] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 507–517. https://doi.org/10.1145/2872427.2883037

[12] Damien Lefortier, Adith Swaminathan, Xiaotao Gu, Thorsten Joachims, and Maarten de Rijke. 2016. Large-scale Validation of Counterfactual Learning Methods: A Test-Bed. (Dec. 2016). arXiv:cs.LG/1612.00367 http://arxiv.org/abs/1612.00367

[13] Daryl Lim, Julian McAuley, and Gert Lanckriet. 2015. Top-N Recommendation with Missing Implicit Feedback. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15)*. ACM, New York, NY, USA, 309–312. https://doi.org/10.1145/2792838.2799671

[14] Harald Steck. 2010. Training and Testing of Recommender Systems on Data Missing Not at Random. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '10)*. ACM, New York, NY, USA, 713–722. https://doi.org/10.1145/1835804.1835895

[15] Adith Swaminathan and Thorsten Joachims. 2015. Batch Learning from Logged Bandit Feedback through Counterfactual Risk Minimization. *Journal of Machine Learning Research* 16 (2015), 1731–1755. http://jmlr.org/papers/v16/swaminathan15a.html

[16] Cai-Nicolas Ziegler, Sean McNee, Joseph A Konstan, and Georg Lausen. 2005. Improving Recommendation Lists through Topic Diversification. In *Proceedings of the 14th International Conference on World Wide Web*. ACM, Chiba, Japan, 22–32. https://doi.org/10.1145/1060745.1060754