5-2019

Studying and handling iterated algorithmic biases in human and machine learning interaction.

Wenlong Sun
University of Louisville

STUDYING AND HANDLING ITERATED ALGORITHMIC BIASES IN HUMAN AND MACHINE LEARNING INTERACTION

By

Wenlong Sun
M.S. Anhui University of Technology, China, 2012
B. Tech., Anhui University of Technology, China, 2009

A Dissertation
Submitted to the Faculty of the
J.B. Speed School of Engineering of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy in Computer Science and Engineering

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, Kentucky

May 2019

STUDYING AND HANDLING ITERATED ALGORITHMIC BIASES IN HUMAN AND MACHINE LEARNING INTERACTION

By

Wenlong Sun
M.S. Anhui University of Technology, China, 2012
B. Tech., Anhui University of Technology, China, 2009

A Dissertation Approved On

January 11, 2019

by the following Dissertation Committee:

Olfa Nasraoui, Ph.D., Dissertation Director

Nihat Altiparmak, Ph.D.

Hichem Frigui, Ph.D.

Roman Yampolskiy, Ph.D.

Scott Sanders, Ph.D.

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Olfa Nasraoui for her endless support and encouragement. Without her patience, motivation and guidance in all aspects of being a good research scientist, this experience would not have been the same.

I would like to extend my gratitude to the rest of my committee members: Dr. Nihat Altiparmak, Dr. Hichem Frigui, Dr. Roman Yampolskiy, and Dr. Scott Sanders for their valuable and constructive feedback. I also want to thank you for letting my defense be an enjoyable moment, and for your important comments and suggestions, thanks to you.

I wish to express my thanks to the Computer Science and Computer Engineering Department and Speed School of Engineering and School of Interdisciplinary and Graduate Studies for providing me the opportunity to pursue my degree and for their unfailing support and assistance throughout my graduate studies and I am grateful for being a recipient of the prestigious Grosscurth Fellowship.

My sincere thanks goes to my fellow lab-mates, with a special mention to Behnoush Abdollahi and Gopi Chand Nutakki. Our conversations on research and their good-hearted support and friendship will not be forgotten.

Words can not express how grateful I am to my mother, my sisters and my brothers for all of the sacrifices that you‘ve made on my behalf. I would also like to thank to my beloved wife, Shufen Wang. Thank you for supporting me for everything, and especially I can‘t thank you enough for encouraging me throughout this experience.

This work was partially supported by National Science Foundation: NSF INSPIRE (IIS)- Grant # 1549981.

ABSTRACT

STUDYING AND HANDLING ITERATED ALGORITHMIC BIASES IN HUMAN AND MACHINE LEARNING INTERACTION

Wenlong Sun

May, 2019

Algorithmic bias consists of biased predictions born from ingesting unchecked information, such as biased samples and biased labels. Furthermore, the interaction between people and algorithms can exacerbate bias such that neither the human nor the algorithms receive unbiased data. Thus, algorithmic bias can be introduced not only before and after the machine learning process but sometimes also in the middle of the learning process. With a handful of exceptions, only a few categories of bias have been studied in Machine Learning, and there are few, if any, studies of the impact of bias on both human behavior and algorithm performance. Although most research treats algorithmic bias as a static factor, we argue that algorithmic bias interacts with humans in an iterative manner producing a long-term effect on algorithms' performance.

Recommender systems involve the natural interaction between humans and machine learning algorithms that may introduce bias over time during a continuous feedback loop, leading to increasingly biased recommendations. Therefore, in this work, we view a Recommender system environment as generating a continuous chain of events as a result of the interactions between users and the recommender system outputs over time. For this purpose, In the first part of this dissertation, we employ an iterated-learning framework that is inspired from human language evolution to study the impact of interaction between

machine learning algorithms and humans. Specifically, our goal is to study the impact of the interaction between two sources of bias: the process by which people select information to label (human action); and the process by which an algorithm selects the subset of information to present to people (iterated algorithmic bias mode). Specifically, we investigate three forms of iterated algorithmic bias (i.e. personalization filter, active learning, and a random baseline) and how they affect the behavior of machine learning algorithms. Our controlled experiments which simulate content-based filters, demonstrate that the three iterated bias modes, initial training data class imbalance, and human action affect the models learned by machine learning algorithms. We also found that iterated filter bias, which is prominent in personalized user interfaces, can lead to increased inequality in estimated relevance and to a limited human ability to discover relevant data.

In the second part of this dissertation work, we focus on collaborative filtering recommender systems which suffer from additional biases due to the popularity of certain items, which when coupled with the iterated bias emerging from the feedback loop between human and algorithms, leads to an increased divide between the popular items (the haves) and the unpopular items (the have-nots). We thus propose several debiasing algorithms, including a novel blind spot aware matrix factorization algorithm, and evaluate how our proposed algorithms impact both prediction accuracy and the trends of increase or decrease in the inequality of the popularity distribution of items over time.

Our findings indicate that the relevance blind spot (items from the testing set whose predicted relevance probability is less than 0.5) amounted to 4% of all relevant items when using a content-based filter that predicts relevant items. A similar simulation using a real-life rating data set found that the same filter resulted in a blind spot size of 75% of the relevant testing set.

In the case of collaborative filtering for synthetic rating data, and when using 20 latent factors, Conventional Matrix Factorization resulted in a ranking-based blind spot (items whose predicted ratings are below 90% of the maximum predicted ratings) ranging between 95% and 99% of all items on average. Both Propensity-based Matrix Factorization

methods resulted in blind spots consisting of between 94% and 96% of all items; while the Blind spot aware Matrix Factorization resulted in a ranking-based blind spot with around 90% to 94% of all items. For a semi-synthetic data (a real rating data completed with Matrix Factorization), Matrix Factorization using 20 latent factors, resulted in a ranking-based blind spot containing between 95% and 99% of all items. Popularity-based and Poisson based propensity-based Matrix Factorization resulted in a ranking-based blind spot with between 96% and 97% if all items; while the blind spot aware Matrix Factorization resulted in a ranking-based blind spot with between 92% and 96% of all items.

Considering that recommender systems are typically used as gateways that filter massive amounts of information (in the millions) for relevance, these blind spot percentage result differences (every 1% amounts to tens of thousands of items or options) show that debiasing these systems can have significant repercussions on the amount of information and the space of options that can be discovered by humans who interact with algorithmic filters.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Websites and online services offer large amounts of information, products, and choices to people. This large amount of information is only useful to the extent that people can find what they are interested in. Researchers have proposed many ways to help users discover what they like. However, all existing approaches aid people by suppressing information that is determined to be disliked or not relevant. Those approaches are considered as an information filter, which allows certain type of information going through, while keeps others away from people. Thus, all of these methods, by gating access to information, have potentially profound implications for what information people can and cannot find, and thus what they see, purchase, and learn.

There are two major adaptive paradigms to help sift through information, *information retrieval* and *recommender systems*. Information retrieval techniques [1–7] have given rise to the modern search engines which return relevant results, following a user's explicit query. For instance, in the probabilistic retrieval model [2], optimal retrieval is obtained when search results are ranked according to their relevance probabilities, which has profound effects on how users find relevant items.

Recommender systems (RS), on the other hand, generally do not await an explicit query to provide results [8–20], consisting of recommend items which are believed to be of interest to the users. Recommender systems can be categorizeed based on which data they use and how they predict user interests. The first type is content-based filtering (CBF) algorithms [10,21,22], which rely on item attributes or user demographics, but often not relations between users (i.e. social relations), as input data. Collaborative Filtering

(CF) [8, 15, 23–26], on the other hand, does not require item attributes or user attributes. Rather it makes predictions about what a user would like based on what other similar users liked. Both adopt algorithms, e.g. K-nearest neighbors [27, 28] and non-negative matrix factorization (NMF) [16, 29–31], that have close analogs in the psychology literature on concept learning, e.g. exemplar models [32–34] and probabilistic topic models [35, 36].

Information filtering algorithms [9, 37, 38] similarly provide users with a list of relevant results, but do so in response to a query. One classic example is the Rocchio filter [39–41], which modifies the user's initial query after a first iteration of search to help filter less relevant results. The query is modified based on the set of initial search result documents which are labeled by the user as relevant and non-relevant, respectively. The new query (which is treated like a pseudo-document) is modified by adding and subtracting a weighted combination of relevant and non-relevant documents, respectively. This is quite similar to content-based recommendation, where information about the items is used to rank potentially relevant results. The process of interaction between query and information filtering algorithm output forms a cyclical feedback loop.

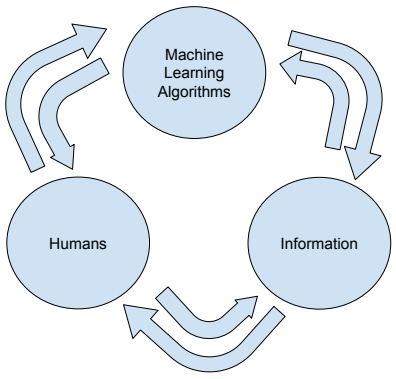


Figure 1.1: Illustration of the interaction between humans, machine learning algorithms, and information. Information is provided in the form of both data from the user to help algorithms learn human preferences and recommendations from the algorithm to help reduce information overload and guide the user.

Common to both recommender systems and information filters is: (1) selection, of a subset of data about which people express their preference, by a process that is not random sampling, and (2) an iterative learning process in which people's responses to the

selected subset are used to train the algorithm for subsequent iterations. The data used to train and optimize performance of these systems are based on human actions. Thus, data that are observed and omitted are not randomly selected, but are the consequences of people's choices. Recommendation systems suggest items predicted to be of interest to a user (e.g. movies, books, news) based on their user profile [10, 11, 24]. The prediction can be based on people's explicit (e.g. ratings) or implicit (e.g. their browsing or purchase history) data [42–45], or even query patterns [46]. Research into human choice suggests that both explicit and implicit choices systematically vary based on context, especially the other options that are present when choosing [47–49].

In addition to the simple effects of the interaction between algorithms' recommendations and people's choices, people may reason about the processes that underlie the algorithms. Research in cognitive science has shown that people reason about evidence selected by other people. In [50], a computational framework was proposed for modeling how people's inferences may change as a consequence of reasoning about why data were selected. This framework has been formalized in learning from helpful and knowledgeable teachers [51–54], deceptive informants [55], and epistemic trust [56–58]. People's reasoning about the intentional nature of the algorithms may exacerbate the effects of cyclic interaction between the algorithms' recommendations and people's choices.

Figure 1.1 illustrates the scope of this work. Machine learning algorithms are affected by information flowing into it and human action. On the other hand, algorithms choose data to use for their training and affect how the information is presented to users. Finally, humans interact with algorithms via their action after consuming the information provided from the algorithms.

We propose a framework for investigating the implications of interactions between human and algorithms, that draws on diverse literature to provide algorithmic, mathematical, computational, and behavioral tools for investigating human-algorithm interaction. Our approach draws on foundational algorithms for selecting and filtering of data from computer science, while also adapting mathematical methods from the study of cultural

evolution [59–61] to formalize the implications of iterative interactions.

Key to our approach is the focus on the sources and consequences of bias in data collected "in the wild". The two primary sources of bias are from algorithms and from humans. Algorithms, such as recommender systems, necessarily filter information with the goal of presenting humans with typically the most preferred content. Then, based on the labels provided by people, learning algorithms are trained to optimize future recommendations. This framework differs from standard learning theory in that the training data are not randomly sampled, which calls into question any guarantees about learning from such data. The second source of bias is people. In addition to receiving iterated information optimized to their preferences, people are also not required to provide labels for any of the presented data. Moreover, people's choices are highly non-random, and may reflect not only their opinions about the presented content, but also inferences about why the content was presented. Finally, bias introduced into the data at any point may be magnified by retraining of models and associated implications for recommendations, yielding algorithms whose performance is at variance with theoretical expectations. We argue that either of the individual sources of bias is in principle sufficient to yield instability, and that this suggests the need for new theories and methods for understanding performance of such systems in terms of human-algorithm interactions. We propose to characterize the conditions under which we would expect these to lead to systematic bias in the selection of information by algorithms, and identify conditions under which we can "undo" the effects of these biases to obtain accurate estimates from biased data. We expect the results to contribute insights back to the fundamental psychology of human reasoning, choice and learning and the fundamental computer science of learning, recommendation and information filtering.

1.2 Problem Statement

In our proposed work, we aim to build a framework for investigating the implications of interactions between humans and algorithms in terms of bias. Five major questions will be answered in this work:

1) How can we build a model to simulate the interaction between humans and algorithms, and how does selection bias (iterated algorithmic bias) in machine learning systems affect algorithm performance over time?

2) How different initializations of learning algorithm lead to different trends of learning given the iterated algorithmic bias modes?

3) How human action (willing to label data whenever requested to) affects algorithm performance over time?

4) How can we measure the impact of different iterated algorithmic bias modes on algorithm performance.

5) How can we debias iterated bias in a recommender system.

1.3 Research Contributions

Previous research treated algorithmic bias as a static factor which fails to capture the dynamic and iterative properties of bias that evolve as a result of human-machine interaction. Inspired by language learning, we developed an *iterated algorithmic bias* model, which considers algorithmic bias as a continuously evolving factor connecting humans and machine learning algorithms through continuous interaction.

In this work, we first present a preliminary theoretical model and analysis of the mutual interaction between humans and algorithms. We also define the concepts of *blind spots* for analyzing the impact of the evolution of bias through iterated learning. To study this impact, we formulate several research questions regarding the effect of iterated algorithmic biases on the behavior of the iterated learning algorithms. Our research questions aim to better understand whether and how iterated algorithmic bias can have an impact on the behavior of simple relevance prediction algorithms.

Secondly, we argue that a recommender system is a continuous chain of events, in which users actively interact with the output of a recommender system. We propose several debiasing algorithms for recommender systems, particularly those based on Matrix Factorization, during this chain of events. More specifically, we propose three algorithms: 1) a

unified recommendation and active learning strategy (active recommendation) used during the interaction between users and a RS algorithm, which tries to reduce uncertainty during recommendations, while keeping good algorithm performance; 2) an exposure-based collaborative filtering recommendation model that is also combined with active recommendation to further debias the recommendation system; and 3) a blind spot aware matrix factorization algorithm, which takes into account the blind spot when learning how to predict recommendations.

1.3.1 Study of Bias in Iterated Learning

The first part of the problem stated in section 1.2 is answered by developing a theoretical and simulation framework for studying bias evolution in interactive learning and elaborating several research questions, including:

(RQ 1): How does class imbalanced initialization affect learned boundaries?

(RQ 2): How do different iterated algorithmic biases have different effects on the behavior of models learned by a ML algorithm (with human action probability equal to 1)? We consider three aspects of a learned model to measure the outcome algorithm's bias:

- **1) Boundary shifts between pre and post iteration (Eq. 3.22);**
- **2) Gini coefficient on predicted testing set labels (Eq. 5.10);**
- **3) blind spot between pre and post iteration (Eq. 3.21).**

(RQ 3): How does class imbalanced initialization affect the model learned during iterative learning? We consider the same three aspects to measure the effects as in RQ 1.

1.3.2 Study of Human Algorithm Interaction

The second part of the problem stated in Section 1.2 is addressed by studying how humans' willing to label the points will affect the algorithm performance.

(RQ 4): How does human action (whether to label data when requested to by the machine learning algorithm) affect the boundary shift? We consider the same three aspects to measure the effects as in RQ 1.

1.3.3 Study of Debiasing Recommender Systems

Recommender Systems (RSs) are widely used to help online-users discover products, books, news, music, movies, courses, restaurants, etc. One major problem with RSs is that they may introduce biases during the continuous feedback loop that occurs with users, and this may lead the RS to make increasingly biased recommendations over time. In this work, we view a RS environment as generating a continuous chain of events that are the result of interactions between users and the RS. Based on this model, we propose several debiasing algorithms during this chain of events, and then evaluate how these algorithms impact the predictive accuracy of the RS, as well as trends in the popularity distribution of items over time. We also propose a novel blind spot awareness matrix factorization (MF) to debias the RS. Results show that the proposed algorithms successfully debias the RS.

1.4 Dissertation Outline

The document started with an introduction to recommendation systems, active learning, iterated learning and bias in Chapter 1. We review additional related work in Chapter 2. Chapter 3 presents a formal model that can help study the problems being researched and pave the way toward studying the impact of iterated bias experimentally. Chapter 4 presents experimental results for the study of iterated algorithmic bias in human and machine learning interaction. Chapter 5 presents research on debiasing recommender systems within an iterative human and machine learning algorithms interaction. Finally, Chapter 6 presents the conclusion of our research and future work.

CHAPTER 2

BACKGROUND

2.1 Recommendation Systems

The large amount of information available on the Web has increased the difficulty of finding what we really want. Also people often need to make choices without enough personal experience with the related items. Recommendation systems are developed to assist and help this process. Sometimes, we get recommendations from other people either by words, online reviews, or some survey results.

Many approaches embody recommender systems as a way of personalizing their content for users. They have the effect of guiding users in a personalized way to interesting objects in a large pool of possible items [62]. Typically, there are three main families of recommendation algorithms: Collaborative filtering (CF), Content-based filtering (CBF), and Hybrid approaches (combining CF and CBF).

2.1.1 Collaborative-Filtering Recommendation Systems

The technique that is most widely used in recommendation systems (RS) is collaborative filtering [63]. In collaborative-filtering, items are recommended to a particular user when other similar users also prefer those items. Similar users may be defined as users having similar ratings on items or users having liked similar items. A collaborative filtering system collects all information about a user's interest on the items and calculates the similarity among the users interests. Users with similar Interests will be clustered into the same group.

Collaborative-Filtering relies only on ratings generated by the users on items [64]. The system recommends to the targeted customer items, which have been rated by other

people, whose ratings are similar to the ratings of the targeted user. The system needs to collect the profile of each user, which can be the set of rated items from that user, the rating can be a Boolean or real number based on different applications. Collaboration filtering searches for similar users (nearest neighbors) or group of users (nearest neighborhood) and then uses ratings from this set of users (groups of users) to predict items that will be liked by the current user [65].

Matrix factorization (MF) is one of the successful CF techniques. MF approximates the rating, r_{ij} given by user i on item j using a factorization so that $r_{ij} \approx p_i q_j$. To solve for p_i and q_j, different approaches can be used to minimize the error between r_{ij} and $p_i q_j$, such as stochastic gradient descent [30]. Many practical approaches have been developed. Singular value decomposition (SVD) is another form of matrix factorization, which usually is considered as baseline to compare other methods [66]. In SVD, the aim is to obtain a factorization of $A = U\Sigma V^T$. The SVD decomposition is to find a lower dimensional feature space. Probabilistic Matrix Factorization (PMF) is a probabilistic linear model with the assumption that the ratings follows a Gaussian distribution with noise [67].The user-item matrix is represented as the product of a lower-rank user feature matrix and an item feature matrix in Probabilistic Matrix Factorization (PMF). It defines the conditional distribution over the observed ratings and latent user-feature matrix, item-feature matrix.

$$p(R|U, V, \delta^2) = \prod_{i=1}^{N} \prod_{j=1}^{M} [N(R_{ij}|U_i^T V_j, \delta^2)]^{I_{ij}}$$

$$p(U|\delta_U^2) = \prod_{i=1}^{N} N(U_i|0, \delta_U^2 I)$$

$$p(V|\delta_V^2) = \prod_{j=1}^{M} N(V_j|0, \delta_V^2 I)$$

where $N(.)$ is the probability density function of the Gaussian distribution. I_{ij} is the indicator function that is equal to 1 if user i rated item j and equal to 0 otherwise. $p(U|\delta_U^2)$ and $p(U|\delta_V^2)$ are the probability distributions for latent space U and V , respectively.

Collaborative filtering recommendation systems have been in existence for a long time in the academic and the industry environments. Electronic mail was one of the first

areas where CF was used and the system was called Tapestry, which is also known as the first recommendation system [8]. CF can also be based on implicit usage data such as web click streams. For example, Nasraoui et al. used a fuzzy approximation reasoning method to build an intelligent web recommendation system [68]. They extracted the user profiles using web usage mining of implicit user interest data and applied clustering algorithms to group the user information in the user database.

2.1.2 Content-Based Recommendation Systems

Content-based recommendation systems recommend an item to a user based on the description of the item and user's interests. Since the user's interests are described in the user's profile, this recommendation algorithm is usually learned from the user's profile and his/her feedback from previous items [21]. In Content-based recommendation, filtering is done based on users' preferred items. In this technique, the items are recommended based on the user and item database. In that database, different items are added from what a user has used in the previous times based on user's personal preferences. Based on the database, user's data files can be constructed according to question-query format, item ratings, or the user's navigation information, which shows the users' potential interest. With content-based recommendation techniques, systems can be built mainly based on the available database and past experience of the users. The disadvantage of this method is that only part of the users give ratings properly [69]. Several algorithms have been developed to learn user profiles, along with the informative description of the items involved.

Item description is very important when using content-based recommendation systems. In many domains, data are not well structured, thus the first step for content-based recommendation systems is to convert the data to a well structured representation. For example, unrestricted text data are presented using stemming at the very beginning [70]. The purpose of stemming is to generate terms which really reflects the common meaning behind words such as "classification", "classifier", "classify", and "classifies". TF-IDF (term-frequency times inverse document frequency) is widely used to represent words in

documents. The TF-IDF weight of a term t in a document d is a function which calculates the weight based on 1) the frequency of t in document d, the number of documents which contains the term t, and finally the total number of documents in the collection. Mooney et al. proposed content-based recommendations of books using a Naive Bayes text classifier [71].

Content-based recommendation systems have a "Profile Learner" module as well. This part collects data representing users' interests in order to construct the user profiles. For example, a web page recommender could implement a relevance feedback method, in which it can learn the combination of vectors of positive and negative examples into a prototype vector inferring the user profile [70].

An obvious advantage of content-based filtering algorithms is that the system does not require domain knowledge. it is sufficient to collect clear feedback from the users about their preferences. This would make content-based filtering the preferred algorithm in situations where explicit ratings from users are difficult to collect. A second advantage is that content-based filtering algorithms have better performance at finding topically similar items than CF algorithms, since CBF systems mainly focus on content similarity (such as the text description of items). When content information is large and complex, it is sometimes difficult to analyze content data for the purpose of recommendations, for example movies reviews and music.

2.1.3 Hybrid Recommendation Systems

Collaborative filtering requires a large historical data set, it also suffers from the cold-start problem. Content-based recommendation systems need detailed profile information on both users and items. Both techniques thus have their respective disadvantages. Hybrid recommendation techniques aim to gain better performance with fewer of the drawbacks of any individual technique [63].

Several combination methods have been proposed over the years. P-Tango system used a weighted recommendation system, in which the ranking score of an item is calculated

by all available recommendation techniques used in this system [72], then it combined these scores using a linear model. Smyth and Cotter proposed a mixed recommendation system for television viewing, in which they applied content-based techniques to the textual description of the TV show, and collaborative filtering on the interest of other users [73]. Another way of combination is to consider the collaborative information as extra feature data associated with each example, and then apply content-based techniques on this new data set. [12] developed a feature combination system for a rule learner in which it used both user ratings and content features for recommending movies, and got significant improvement in accuracy. Inspired by weighted recommendation systems, researchers developed meta-level combination, in which one recommendation technique's output is used as the input to another technique. Balabanovic built a web filtering system with such a meta-level combination [74].

Other methods of combination include cascade and feature augmentation. In the cascade technique, one recommendation technique is applied first to generate a ranking of the candidate, then a second technique is used to refine the recommendation [63]. For feature augmentation, the first technique is used to produce a rating of an item, then the information of this item is combined into the processing of the next recommendation technique.

2.1.4 User Feedback

Recommendation systems have a natural property involving human action in terms of feedback. For example, people interact with the output of recommendation system, leading to the impact in both ways : 1) The recommendation systems show only some 'filtered items' or outputs to the user; 2) The user tends to label or rate those outputs only (in the strict case). Studies have shown that the interaction between users and recommendations will lead to a well-known phenomenon named 'filter bubble' [75], and it will finally decrease the user satisfaction about recommender systems slowly [76]. It is also important to notice that assuming that the inner relation between users and items is not changing over time may lead

to users seeing only a narrow part of the entire set of options available to recommend [77].

Generally, user feedback in recommendation systems is classified as either explicit feedback which contains ratings by users regarding their preference in products, or implicit feedback such as clicks or purchase [76]. A considerable amount of work on integrating the information from the two types of user feedback mechanisms in recommendation system in order to improve and personalize recommendations are performed [14, 30, 75]. Here, we do not focus on improving the accuracy of recommender systems, but instead, we are trying to understand how the interaction will change people' ability to discover new items. Another way of looking at this is to find the exploration and exploitation trade-off in filtering algorithms.

2.2 Active Learning

Active learning (sometimes called "query learning" in statistical literature) is usually described compared with passive learning, which is sometimes referred to as supervised learning and aims to build up an accurate predictor from labeled training data [78]. A passive learner receives a random data set and learns a classifier as output. However, in many cases labeling instances is money and time consuming. In the other hand, a large pool of unlabeled samples are easy to get. Now, instead of choosing random samples to be manually labeled for training, the algorithm can interactively query the user to obtain the desired data sample to be labeled [79]. Generally, there are five query approaches :

- Uncertainty sampling: In this approach, an active learner queries the users with the sample which is least certain to label. This approach tries to label those points for which the current model is least certain [80]. For three or more class labels, a general uncertainty model is:

  $$x^* = argmax_x(1 - P_\theta(\hat{y}|x))$$

  where $\hat{y} = argmax_y P_\theta(y|x)$ is the class label with the highest posterior probability under the model θ.

- Query by committee: Several models are trained on the current labeled data then vote on the output for unlabeled data [81]. This strategy maintains a committee $C = \{\theta^{(1)}, ..., \theta^{(c)}\}$ of different models, which are all trained using the labeled set, but indicating contradictory hypotheses. Then each committee member can vote on the labeling of query hypotheses.

- Expected model change: This approach uses a decision-theoretic approach, selecting the instance that would impart the greatest change to the current model if we knew its label, and label those points which would most change the current model [82].

- Expected error reduction: Compared with the Query-by-committee approach, this approach aims to measure how much its generalization error is likely to be reduced instead of how much the model will change. Basically it tries to measure the expected future error of the model trained on those remaining unlabeled instances, and labels those points which would most reduce the model's generalization error [82]. .

- Variance reduction: This approach tries to reduce generalization error indirectly by minimizing the output variance, which has a closed-form solution, and labels those points that would minimize the output variance, which is one of the components of error [83].. A learner's expected future error can be written as:

  $$E_T[(\hat{y} - y)^2|x] = E[(y - E[y|x])^2] + (E_L[\hat{y}] - E[y|x])^2 + E_L[(\hat{y} - E_L[\hat{y}])^2]$$

  Here $E_L[.]$ is an expectation over the labeled set. $E[.]$ is an expectation over the conditional density $P(y|x)$, and E_T is an expectation over both.

Active Learning has been widely used for decades. Simon Tang proposed a new algorithm for performing active learning with SVM, which took advantage of the duality between the parameter space and feature space and significantly reduced the need for labeled training instances [84]. In text classification, most of the active learning algorithms mainly select a single unlabeled document in each iteration, Steven Hoi developed a novel approach which selected a batch of text documents for labeling manually in each iteration. The author performed experiments on large-scale text categorization, and results were convincing [85].

Active learning is also used in speech recognition [86–88]. Sequence labeling, in natural language processing is extremely time consuming, Burr Settles presented several query strategies for probabilistic sequence models for natural language processing [86]. The criteria to select queries is to find a way to assess how informative each instance is. Zhang et al. [87] modified several query models for sequences and proposed an active learning framework for content-based information retrieval, it was tested on image retrieval from a database. Most commonly, the approach to collect data in active learning is to select data points that are close to the boundary. Spoken dialog systems aim to identify the intended meaning of human utterances. The intent of the speaker is identified from the recognized sequence by a spoken language understanding system (SLU). This task can be considered as a classification problem. Gokhan Tur [88] combined active and semi-supervised learning for a better understanding of spoken language.

In recommendation systems and information retrieval, active learning is also used widely. Recommendation systems aim to present a particular item to user: to learn more about user's preferences, or likes and dislikes. Then active learning naturally comes in, since user can help the recommendation system to get new information about the system [89]. Most commonly, the approach to collect data in active learning is to select data points close to the boundary. Hieu T. Nguyen [75] built a framework which incorporated clustering into active learning. The algorithm first constructed a classifier on cluster representatives, and then propagated the classification decision to the other samples via a local noise model.

2.3 Iterated Learning and Language Evolution

In language learning, humans form their own mapping rules after listening to others, and then speak the language following the rules they learned, which will affect the next learner.

Researchers have shown that iterated learning can produce meaningful structure patterns in language learning [90–94]. In particular, the process of language evolution can be viewed in terms of a Markov chain, as shown in Figure 2.1. The first learner sees some

linguistic data and forms a hypothesis, then the learner produces their own data, which is the input to the next learner. After enough iterations, the hypothesis emerged from this process becomes certain; we should expected an iterated learning chain to converge to the prior distribution of all hypotheses given that the learner is a Bayesian learner [59]. That is, the knowledge learned is not accumulated during the whole process. We refer to this iterated learning model as *pure iterated learning (PIL)*. One problem about this iterated language learning model is that it is difficult practically to prove the convergence or the boundary, even if it is proved theoretically [92]. Rafferty et al. gave an upper bound about the convergence, saying the convergence occurs in a number of generations that is $nlog(n)$ for Bayesian learning of the ranking of n constraints or the n binary parameters values [95].

Extension from previous work of iterated learning on language has also attracted attention from researchers. Perfors et al. revealed that the learners will converge to languages that depend on the structure as well as their prior biases when certain assumptions about the independence of language and the world are abandoned [96]. In Figure 2.1, there is no dependency between current input x and the previous learned hypothesis, which represents the graphic model of PIL.

Iterated learning, on the other hand, is similar to the online learning process [97]. However, they are different in several ways. Online learning appears in a consecutive manner. The learner is required to provide an answer to a given question on each round. To answer the question, the learner uses a prediction model, or a hypothesis, which maps from sets of questions to the set of answers. After prediction, the quality is measured based on the true answers. The goal is to minimize the cumulative loss in the online learning process [98]. Meanwhile, in iterated learning, we are interested in investigating the given information's effect on the learned hypothesis.

2.4 Relationship between Iterated Learning and Information Retrieval

It is interesting to recognize how Iterated Learning manifests itself in the context of adaptive information filters, as exemplified by modern search engines. Based on infor-

mation retrieval, modern search engines return relevant results, following a user's explicit query [7]. For instance, in the probabilistic retrieval model, optimal retrieval is obtained when search results are ranked according to their relevance probabilities [2]. Recommender systems, on the other hand, generally do not await an explicit query to provide results [14]. Both Information retrieval and recommender system, to some extent, require information selection to get better results, thus iterated interactive learning naturally fits the purpose of studying the interaction of algorithms and humans.

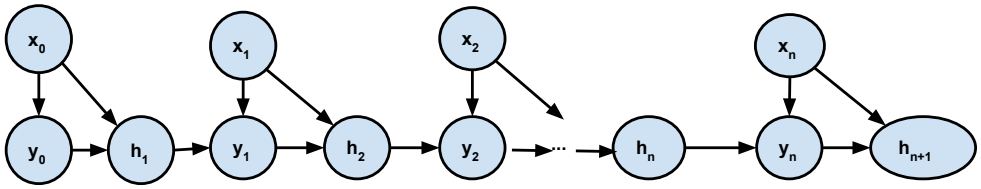


Figure 2.1: Illustration of iterated learning with(bottom)/without(top) dependency from previous iterations. In iterated learning, information is passed through selected data, where the inputs, x, are independent of the inferred hypothesis.

2.5 Bias in Machine Learning

Studying bias within the machine learning context is not new. According to Mitchell [99], bias is "any basis for choosing one generalization over another, other than strict consistency with the instances." Gordon extended this concept to include any factor which affects the definition or selection of inductive hypotheses [100]. Bias is used in some well-known machine learning algorithms. For example, decision-tree algorithms have a bias for simple decision trees, while rule induction algorithms have a bias for simple disjunctive normal form (DNF) expressions, neural-network methods have a bias for linear threshold functions, and finally naive Bayes can be considered to have a bias for functions that assume conditional independence between features [71]. The fit between a particular bias and a problem generally affects the performance of a machine learning algorithm on that problem.

There are mainly two types of bias in machine learning: 1) representational bias

and 2) procedural bias. As quoted in [100] "representational bias characterize the search space in machine learning algorithms". Typically, those search spaces are the spaces of the hypotheses. A procedural bias (also called algorithmic bias) decides the order of traversal of the states in the space defined by a representational bias [101]. Examples of procedural biases include the beam width in a beam search and a preference for simple or specific hypotheses. Both representational and procedural biases can be assessed by determining the effect they have on learning performance.

Algorithmic bias can be categorized based on the time in which it occurs during the machine learning process [102]. Generally, selecting biased training samples leads to a biased model [103]. Training data bias may come from various sources which depend on the application, such as human labeling, sample selection and others. Several machine learning techniques have been proposed to deal with this problem [104–110]. Charles Elkan proposed to build a more economically coherent cost matrix when dealing with the classification problem in the economical domain [104]. Zadrozny formalized the sample selection bias problem in a number of well-known classifier learning methods and studied how they are affected the sample selection bias [105]. She categorized learning algorithms into 'global learner' and 'local learner' based on how the boundary are affected by the selected biased samples. Bianca Zadrozny proposed a re-weighting strategy based on cost-proportionate weighting of the training examples, which can be accomplished either by bringing the weights to the classification algorithm (as often done in boosting), or by careful sub-sampling [106]. Miroslav Dudik et al. studied the problem of density estimation under sample selection bias and proposed three bias-correction approaches, which are based on the statistical information of samples [107]

The post-algorithmic bias emerges when users interpret the output of machine learning algorithms [111,112]. Ricardo Baeza-Yates [111] suggests that the most popular knowledge in the web are actually coming from a few users, who are very active. When the users take in those knowledge, they automatically become the victims of algorithmic bias [111]. Dino Pedreshi argued that algorithmic bias has been identified and critiqued for its impact

on search engine results, social media platforms, privacy, and racial profiling [112], and suggested that removing some of the discriminate features from data is not enough. The author introduced a notion of discriminatory classification which they claimed to have better bias correction impact.

Recent research in algorithmic bias has generally focused on the ethical problems that machine learning algorithms might create [113–117]. For instance, Zook et al. [116] have recently argued that researchers must carefully check the impact of algorithms on specific groups of people (such as defined by gender and race) before deploying algorithms. Kirkpatrick [113] illustrated the ethical problems that can occur when algorithmic bias is introduced in the justice system. Garcia [115] stated that algorithmic bias may worsen racist problems in certain circumstances. More recently, Kate Crawford's presentation related to research on the fairness of machine learning algorithms attracted more attention from the machine learning community to the problem of algorithmic bias [116]. Helen Nissenbaum indicated that algorithmic bias occurs when a computer system behaves in ways that catches the implicit values of humans involved in that data collection, selection, or use [118]. Kate Crawford argues that algorithmic bias is getting worse if the learning algorithms are 'black-box' [114]. In all the above discussion of bias [113–118], the bias impact from the interaction between humans and computer systems was not mentioned.

In statistics, bias refers to the systematic distortion of a statistic. Here we can distinguish a biased sample, which means a sample that is incorrectly assumed to be a random sample of a population, and estimator bias, which results from an estimator whose expectation differs from the true value of the parameter [119].

2.6 Related Work on Debiasing Recommender Systems

Work on debiasing RSs has been done from various perspectives. Hu et al. first proposed using an implicit feedback model to measure the level of confidence that a user will see an item [120]. This is similar to the modern notion of a propensity score [121], which is defined as the probability that an item will be seen by the user. Schnabel et

al. [121] argued that recommending items in a RS is analogous to exposing a patient to a randomized treatment in a medical study, and proposed the introduction of a propensity-scored recommendation learning system. Liang et al. proposed that a propensity score matrix be calculated first, followed by a weighted MF based on the propensity score matrix [122, 123]. Badami et al. studied item polarization in recommender system and developed algorithms to counter polarization in recommender system [124]. Abdollahpouri et al. proposed a fairness-aware regularization aiming to reduce popularity bias in recommender system [125]. Chanely et al. made a series of interesting observations about how algorithms increase the homogeneity among users, thereby decreasing the utility gained by users of the system, and they presented a simulation to illustrate how this effect occurs [126]. Yang et al. considered implicit feedback models instead of explicit feedback models in their study, and proposed ways to estimate a propensity matrix based exclusively on popularity, which essentially describes propensity as an estimator of the true probability distribution [127]. Singh et al. proposed a method to construct fair rankings among relevant items, positing that the problem originates at the ranking step (recommendation step) and not during the learning process [128]. Their methodology ranked the items based on a calculated utility, and added a fairness constraint based on the propensity score. Sinha et al. considered the RS as a feedback loop, assumed that user ratings are true prior to the feedback loop, and proposed a method to deconvolve this feedback mechanism, assuming that each feedback response follows a certain mathematical relationship to previous recommendations [129]. O. Nasraoui et al. considered the continuous interaction between learning algorithms and human as a Markov Chain of event, and proposed several approaches to debias the learning algorithms [77]. Shafto et al. studied how the continuous interaction between learning algorithms and human affect behavior patterns of human and proposed possible cognitive intervene to debias the interaction [130].

2.7 Summary and Conclusions

This chapter presented an introduction to collaborative filtering and content-based recommendation systems, and how it can be used as a good way to learn human-algorithm interaction. We then reviewed active learning and its applications as well as biases in machine learning. After that, we gave an introduction on iterated learning and its relationship with language evolution, as well as information retrieval.

CHAPTER 3

ITERATED ALGORITHMIC BIAS IN ONLINE LEARNING

3.1 Introduction

Recommendation systems are becoming increasingly common and powerful nowadays, and are widely used in industry, where websites and Apps recommend items and change the order and the appearance of information on user interfaces to potential users to achieve a variety of goals such as increasing sales, or to assist users in finding information faster. Users can easily get engaged with recommendation systems on social media networks, such as Twitter, and Facebook. Recommendation systems are therefore good tools to study the interaction between human users (the consumers of algorithms) and algorithms. In our proposed work, we will focus on content-based recommendation systems, because the adopted framework studies the interaction as more items are added to the training data used to learn a machine learning model. We describe a methodology to study continuous interaction between humans and algorithms while modeling iterated bias that builds-up during this interaction.

3.2 Iterated Algorithmic Bias

We have reviewed the iterated algorithmic bias in Section 2.5 . which can refer to a wide array of meanings depending on the field and context, ranging from social bias to machine learning bias. Within our scope, bias is closer to the sample bias and estimator bias from statistics, however, we are interested in what we call **'iterated algorithmic bias'** which is the dynamic bias that occurs during the selection by machine learning algorithms of data to show to the user to request labels in order to construct more training data, and subsequently update their prediction model, and how this bias affects the learned (or

estimated) model in successive iterations. Table 3.1 shows the related work and how our proposed framework is different from existing approaches.

Taking all the above in consideration, we observe that most previous research has treated algorithmic bias as a static factor, which fails to capture the iterative nature of bias that is borne from continuous interaction between humans and algorithms. We argue that algorithmic bias evolves with human interaction in an iterative manner, which may have a long-term effect on algorithm performance and humans' discovery and learning.

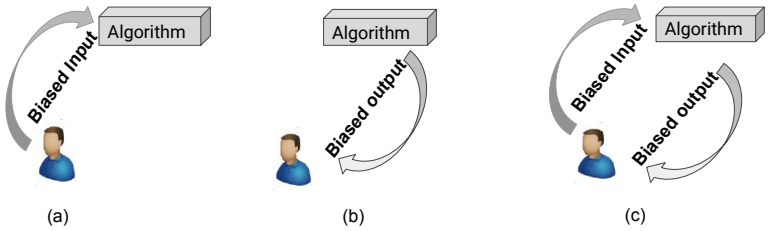


Figure 3.1: Evolution of bias between algorithm and human. In sub-figure (a), biased data from a human may lead to a biased algorithm, this is pre-algorithmic bias. In sub-figure (b), a biased algorithmic output might affect human behavior: For instance, by hiding certain items from humans, algorithms may affect human discovery, learning, and awareness in the long term. Sub-figure (c) indicates a continuous interaction between humans and algorithms that generates bias that we refer to as **iterated bias**, namely bias that **results from repeated interaction between humans and algorithms**.

In this work, we focus on simulating how the data that is selected to be presented to users affects the algorithm's performance and how human choice of action (specifically, to label or not to label the selected instance(s), that are presented to them by the algorithm), may in turn affect the algorithm's performance (see Figure 3.1). In this work, we choose recommendation systems as the machine learning algorithm to be studied. One reason is that recommendation systems have more direct interaction options with humans, while information retrieval focuses on getting relevant information only. Recommendation systems have a natural property involving human action. For example, people interact with the output of recommendation system, leading to the impact in both ways : 1) The recommendation systems show only some 'filtered items' or outputs to the user; 2) The user tends to label or rate those outputs only (in the strict case) (see Figure 3.2). We further simplify

TABLE 3.1

Different bias types in recent research. Iterated algorithmic bias happens when an algorithm interacts with human response continuously, and updates its model after receiving feedback from the human. Meanwhile, the algorithm interacts with the human by showing only selected items or options. Other types of bias are static, which means they have a one-time influence on an algorithm.

Bias type	Iterative	Ethical issue	Pre-algorithm	Post-algorithm	Research
Feature	✗	✓	✓	✗	[113–117]
Human label	✗	✓	✓	✗	[108–110]
Sample selection	✗	✓	✓	✗	[104–107]
Iterated algorithmic	✓	✓	✓	✓	this study

the recommendation problem into a 2-class classification problem, e.g. like/relevant (class 1) or dislike/non-relevant (class 0), thus focusing on a personalized content-based filtering recommendation algorithm.

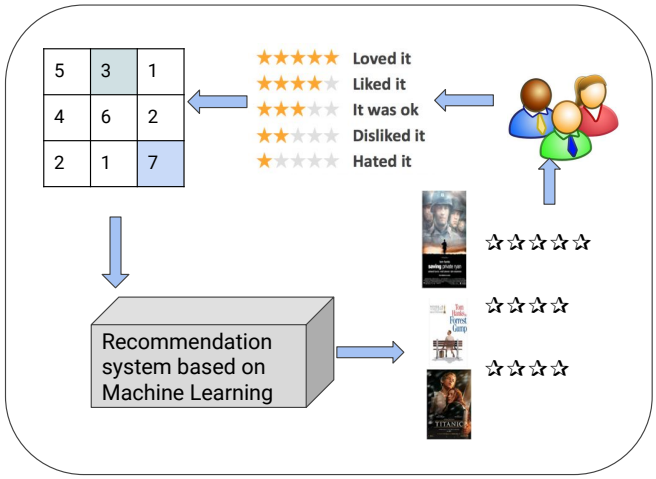
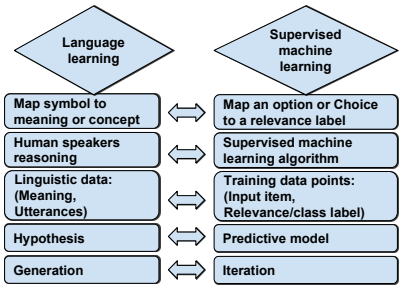


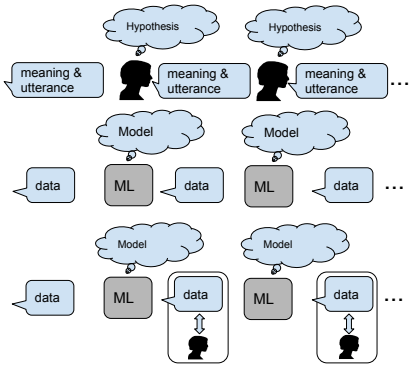
Figure 3.2: Illustration of the feedback loop in recommender system.

3.3 Human-Algorithm Interaction Mechanism

As stated in Section 2.3, language learning and machine learning have several properties in common. Figure 3.3 shows the analogy between language evolution and iterated machine learning. For example, a 'hypothesis' in language is analogous to a 'model' in machine learning. Learning a language which gets transmitted throughout consecutive generations of humans is analogous to learning an online model throughout consecutive iterations of machine learning. Inspired by the language evolution, we adopt this concept into human-algorithm interaction.



(a) Analogy between language learning and machine learning



(b) Chain of events in language learning (top), machine learning (middle) and iterative human-machine learning algorithm interaction (bottom). ML refers to Machine Learning

Figure 3.3: Language learning vs. machine learning. Language learning is analogous to machine learning in several aspects, such as 'hypothesis' to 'model' in sub-figure (a). Learning a language which gets transmitted throughout consecutive generations of human speakers is analogous to learning a model through consecutive iterations of online machine learning in sub-figure (b). In the iterative human-machine learning algorithm interaction, the output from ML affects human behavior and human also interact with the output which affects next iteration.

Because we are interested in studying the interaction between machine learning algorithms and humans, we adopt an efficient way to observe the effect from both sides by using iterated interaction between algorithm and human action. Researchers from behavioral science have developed frameworks for investigating the effect of iterative interactions. It is known that iterated interaction can be considered to generate Markov chains in the long-run, which gives us a well-formed framework to analyze the asymptotic effects of local

decision. As stated before, we consider a simplified recommendation problem consisting of a 2-class classification problem. Thus we start with simple supervised machine learning to predict the 'relevance' class label of an item for a single user.

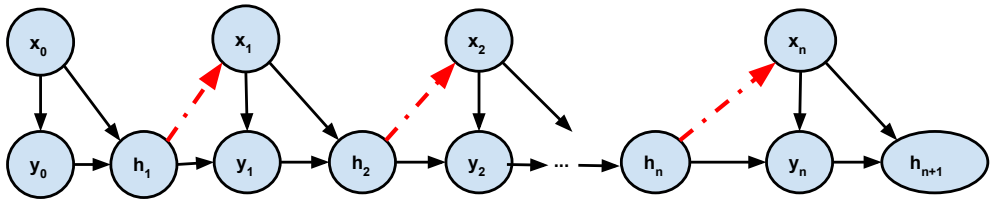


Figure 3.4: Illustration of iterated learning with dependency from previous iterations. In iterated learning, information is passed through selected data, where the next inputs are selected based on the previous hypothesis. This is more consistent with recommender systems and information filtering circumstances.

To begin, we consider three possible mechanisms for selecting information to present to users: **Random, Active-bias, and Filter-bias**. These three mechanisms simulate different regimes. Random selection is unbiased and used here purely as baseline for no filtering. Active-bias selection introduces a bias whose goal is to accurately predict user's preferences. Filter-bias selection brings a bias whose goal is to provide relevant information or preferred items.

Before we go into the three forms of iterated algorithmic bias, we first investigate Pure Iterated Learning (PIL). We adopt some of the concepts from Griffiths [59]. Consider a task in which the algorithm learns a mapping from a set of m inputs $X = \{x_1, ..., x_m\}$ to m corresponding outputs $\{y_1, ..., y_m\}$ through a latent hypothesis h . For instance, based on previous purchase or rating data (x, y), a recommendation system will collect a new data about purchased item (x_{new}, y_{new}) and update its model to recommend more interesting items to users. Here, x represents the algorithm's selections and y represents people's responses (e.g. likes/dislikes). Following Griffiths' model for human learners, we assume a Bayesian model for prediction [131]

Figure 3.4 shows the iterated learning strategy with dependency comparing what is in language evolution. We adapt some of the concepts from Griffiths [131].

3.3.1 Iterated Learning without Dependency

Iterated learning provides a framework to analyze the evolution of the learned hypotheses with subsequent interactions between the user and the algorithm. Without dependencies between variables, the next generated input did not depend on the previous hypothesis learned by the algorithm. In our study, we need to take into account the dependency between the current hypothesis and the next input, because the model or hypothesis learned by the algorithm (learner) is used as a filter to the types of data that will control what data can be seen by the user.

Consider the same task as figure 2.1. Given input data point x, hypothesis h decides a conditional probability of y, i.e. $p(\mathbf{y}|\mathbf{x}, h)$, which connects the inputs and outputs. In the learning step $n+1$, the algorithm sees data point$(\mathbf{x}_n, \mathbf{y}_n)$ and computes a posterior distribution over h_{n+1} using Bayes rule,

$$p(h_{n+1}|\mathbf{x}_n, \mathbf{y}_n) = \frac{p(\mathbf{y}_n|\mathbf{x}_n, h_{n+1})p(h_{n+1})}{p(\mathbf{y}_n|\mathbf{x}_n)} \tag{3.1}$$

where

$$p(\mathbf{y}_n|\mathbf{x}_n) = \sum_{h' \in H} p(\mathbf{y}_n|\mathbf{x}_n, h')p(h') \tag{3.2}$$

and it is assumed that each y_i is independent given x_i and h, so $p(\mathbf{y}|\mathbf{x}, h') = \prod_i p(y_i|x_i, h')$.

With multiple learning iterations, the process will form a Markov chain, in which the posterior distribution at $n+1$ depends on the distribution at n through the prior the distribution $p(h_{n+1})$ and the sampling process $p(\mathbf{y}_n|\mathbf{x}_n, h_{n+1})$. With any Markov chain, the long-run behavior of iterated learning can therefore be considered as transition matrix, $T(h_{n+1}, h_n) = p(h_{n+1}|h_n)$. The probability $p(h_{n+1}|h_n)$ can be calculated as follows:

$$p(h_{n+1}|h_n) = \sum_{\mathbf{x} \in X} \sum_{\mathbf{y} \in Y} p(h_{n+1}|\mathbf{x}, \mathbf{y})p(\mathbf{y}|\mathbf{x}, h_n)p(\mathbf{x}) \tag{3.3}$$

where the equation has been simplified according to the dependency structure in

figure 2.1. The asymptotic behavior of the Markov chain can be derived with the transition matrix $T(h_{n+1}, h_n)$. This chain, being ergodic, converges to a stationary distribution $\pi(h)$ satisfying the equation $\pi(h_{n+1}) = \sum_{h_n \in H} T(h_{n+1}, h_n)\pi(h_n)$.

$$p(h_{n+1}) = \sum_{h_n \in H} p(h_{n+1}|h_n)p(h_n) = p(h_n) \tag{3.4}$$

This means that algorithms that engage in iterated learning will converge to the prior distribution over hypotheses. In other words, if at each step the system randomly selects inputs $\mathbf{x}$, which are then paired with outputs $\mathbf{y}$, and it does not accumulate the data over time, then it will loose all information. Without the dependency, at step $n+1$ we see input $\mathbf{x}_{n+1}$ which is generated from a distribution $p(\mathbf{x})$ that is independent of all other variables.

3.3.2 Iterated Learning with Iterated Filter-bias Dependency

The extent of the departure that we propose from a conventional machine learning framework toward a human - machine learning framework, can be measured by the contrast between the evolution of iterated learning *without* and *with* the added dependency. As shown in Figure 3.4, without the dependency, the algorithm at step n accepts input point x from a set X, which is generated from a distribution $p(x)$ that is independent of all other variables. We used notation $q(x)$ to represent this independence. Here, $q(x)$ indicates an unbiased sample from the world, rather than a selection made by the algorithms. On the other hand, with the dependency, the algorithm at iteration n sees input x_n which is generated from both the objective distribution $q(x)$ and another distribution $p_{seen}(x)$ that captures the dependency on the previous hypothesis h_n which implies future bias of what can be seen by the user. Thus, the probability of input item x is given by:

$$p(\mathbf{x}|h_n) = (1-\epsilon)p_{seen}(\mathbf{x}|h_n) + \epsilon q(\mathbf{x}) \tag{3.5}$$

Recall that the probability of seeing an item is related to its rank in a rating based recommendation system or an optimal probabilistic information filter [2]. For a rating based recommendation system, the ranking is based on the prediction from the system, or the probability of relevance from prediction. In both situations, the selected data point x is

likely to be highly rated or relevant, given h. In most circumstances, the recommendation system has a preferred goal, such as recommending relevant items (with y=1). Then x will be chosen based on the probability of relevance $p(y = 1|x, h_n)$, $x \in X$. Assume that we have a candidate pool X at time n (In practice X would be the data points or items that the system can recommend at time n), then

$$p_{seen}(\mathbf{x}|h_n) = \frac{p(\mathbf{y} = 1|\mathbf{x}, h_n)}{\sum_{\mathbf{x}\in X} p(\mathbf{y} = 1|\mathbf{x}, h_n)} \tag{3.6}$$

The selection of inputs depends on the hypothesis, and therefore information is not unbiased, $p(\mathbf{x}|h_n) \neq q(\mathbf{x})$. The derivations of the transition probabilities in Eq. 3.6 will be modified to take into account Eq. 3.5, and will become

$$p(h_{n+1}|h_n) = \sum_{\mathbf{x}\in X} \sum_{\mathbf{y}\in Y} p(h_{n+1}|\mathbf{x}, \mathbf{y}) p(\mathbf{y}|\mathbf{x}, h_n) p_{seen}(\mathbf{x}|h_n) \tag{3.7}$$

Eq. 3.7 can be used to derive the asymptotic behavior of the Markov chain with transition matrix $T(h_{n+1}) = p(h_{n+1}|h_n)$, i.e.

$$p(h_{n+1}) = \epsilon p(h_{n+1}) + (1 - \epsilon) T_{bias} \tag{3.8}$$

$$T_{bias} = \left[\sum_{\mathbf{x}\in X} \sum_{\mathbf{y}\in Y} p(h_{n+1}|\mathbf{x}, \mathbf{y}) \sum_{h_n \in H} p(\mathbf{y}|\mathbf{x}, h_n) p_{seen}(\mathbf{x}|h_n) \right] p(h_n) \tag{3.9}$$

To illustrate the effects of filter bias, we can analyze a simple and most extreme case where the filtering algorithm shows only the most relevant data in the next iteration (e.g. top-1 recommender). Hence

$$x^{top} = \underset{x}{argmax} P(y|x, h) \tag{3.10}$$

$$p_{seen}(\mathbf{x}|h_n) = \left\{ \begin{array}{l} 1 \, for \, x = x^{top} \\ 0 \quad otherwise \end{array} \right\} \tag{3.11}$$

$$T_{bias} = \left[\sum_{\mathbf{x}\in X} \sum_{\mathbf{y}\in Y} p(h_{n+1}|\mathbf{x}, \mathbf{y}) \sum_{h_n \in H} p(\mathbf{y}|\mathbf{x}_n^{top}, h_n) \right] p(h_n). \tag{3.12}$$

Based on equation 3.7, the transition matrix is related to the probability of item x being seen by the user, which is the probability of belonging to class $y = 1$. The fact that x_n^{top} maximizes $p(y|x, h)$ suggests limitations to the ability to learn from such data. This

limitation of seeing more items causes a well-known problem in language learning study, i.e., 'information bottleneck' [96]. Amy Perfors and Daniel Navarro showed that human learners with Bayes reasoning will converge to the expected posterior distribution over languages given meaningful event in the world [96]. Specifically, the selection of relevant data allows the possibility of learning that an input that is predicted to be relevant is not, but does not allow the possibility of learning that an input that is predicted to be irrelevant is actually relevant. In this sense, **selection of evidence based on relevance is related to the confirmation bias in cognitive science**, where learners have been observed to (arguably maladaptively) select data which they believe to be true (i.e. they fail to attempt to falsify their hypotheses) [132]. **Put differently, recommendation algorithms may induce a blind spot where data that are potentially important for understanding relevance are never seen.**

A major impact of this filter bias is that it shapes what data each leaner will see, which is similar to 'information bottleneck' as stated in [96]. We now prove the convergence of this Markov chains. Let's assume that after long-run with filter bias, the learner will see data x^*, $x^* \in q(x)$. We start with three important assumptions:

- 1) All the learners in the Markov chain follow a Bayesian rule;
- 2) The filter bias affects the data that the learner will see, therefore it changes the structure of the data that the learner consumes;
- 3) The posterior probability of a hypothesis given x is close to its expected posterior probability given the generating distribution $q(x)$ for some x, i.e., $p(h|x) \approx E_{q(x^*)}[p(h|x^*)] = \sum_{x^*} p(h|x^*)q(x^*)$.

Lemma: The stationary distribution of the Markov chain with filter bias is $\pi(h)$, where $\pi(h) = \sum_x p(h|x)q(x)$.

Proof: According to the prove from Griffith [59] and Perfors [96], for the $\pi(h) = \sum_x p(h|x)q(x)$ to be the stationary distribution, the following must be true.

$$
\begin{aligned}
\pi(h_{n+1}) &= \sum_{h_n} p(h_{n+1}|h_n)\pi(h_n) \\
&= \sum_{h_n \in H} \left[\sum_{x \in X} \sum_{y \in Y} p(h_{n+1}|x,y)p(y|x,h_n)q(x) \right] \pi(h_n) \\
&= \sum_{h_n \in H} \left[\sum_{x \in X} \sum_{y \in Y} p(h_{n+1}|x,y)p(y|x,h_n)q(x) \right] \sum_{x^*} p(h_n|x^*)q(x^*) \\
&\approx \sum_{h_n \in H} \left[\sum_{x \in X} \sum_{y \in Y} p(h_{n+1}|x,y)p(y|x,h_n)q(x) \right] p(h_n|x) \\
&= \sum_{x \in X} \sum_{y \in Y} p(h_{n+1}|x,y)q(x) \sum_{h_n \in H} p(y|x,h_n)p(h_n|x) \\
&= \sum_{x \in X} q(x) \sum_{y \in Y} p(h_{n+1}|x,y)p(y|x) \\
&= \sum_{x} p(h_{n+1}|x)q(x) \\
&= \pi(h_{n+1})
\end{aligned} \tag{3.13}
$$

The proof indicates that there is a relationship between h_n and x, here x are meaningful events/points from the generating distribution $q(x)$. It successively shows that the convergence is also affected by the structure of the data.

3.3.3 Iterated Learning with Iterated Active-bias Dependency

Active learning was first introduced to reduce the number of labeled samples needed for learning an accurate predictive model, and thus accelerate the speed of learning towards an expected goal [78, 133]. Instead of choosing random samples to be manually labeled for the training set, the algorithm can interactively query the user to obtain the desired data sample to be labeled [82].

$$
p_{active}(\mathbf{x}|h) \propto 1 - p(\hat{\mathbf{y}}|\mathbf{x}, h) \tag{3.14}
$$

where $\hat{\mathbf{y}} = \arg\max_y (p(y|\mathbf{x}, h))$. That is, $\mathbf{x}$ values are selected to be least certain about $\hat{\mathbf{y}}$, the predicted y value.

Assuming a simplified algorithm where only the very uncertain data are selected, we can investigate the limiting behavior of an algorithm with the active learning bias. Assuming a mixture of random sampling and active learning, we obtain:

$$x^{act} = \arg\max_{x} \left(1 - p(\hat{\mathbf{y}}|\mathbf{x}, h)\right) \tag{3.15}$$

$$p(h_{n+1}) = \epsilon p(h_{n+1}) + (1 - \epsilon) T_{active} \tag{3.16}$$

Where

$$T_{active} = \left[\sum_{\mathbf{x} \in X} \sum_{\mathbf{y} \in Y} p(h_{n+1}|\mathbf{x}, \mathbf{y}) \sum_{h_n \in H} p(\mathbf{y}|\mathbf{x}_n^{act}, h_n) \right] p(h_n). \tag{3.17}$$

The limiting behavior depends on the iterated active learning bias, $\mathbf{x}_n^{act}$. This is, in most cases, in opposition to the goal of filtering, the algorithm will only select data point(s) which are closest to the learned model's boundary, if we are learning a classifier for example. In contrast, the filtering algorithm is almost certain to pick items that it knows are relevant.

This is, of course, consistent with the different goals of recommendation and active learning. The analysis illustrates how the long-run implications of these different biases may be analyzed: By deriving the transition matrices implied by iterated application of data selection biases, we can see that both active learning and filtering have different goals, but focus on an ever more extreme (and therefore not representative) subset of data. Similar methods can be applied to more nuanced and interesting biases to shed light on the consequences of iterative interactions on the data.

3.3.4 Iterated Learning with Random Selection

The iterated random selection is considered as baseline for comparison purpose. This selection mechanism randomly choose instance to pass to next learner during iterations.

3.4 Iterated Learning with Human Action Bias

The above analysis assumes that people's response is always observed. In the following, we extend our analysis to the more realistic case where users have a choice of whether to act or not on a given input.

Assume that people have some target hypothesis, h^*, which represents optimal performance for the algorithm. Data are composed of an input provided by the algorithm, x, an output, y, and an action, a. The indicator variable a takes a value of 1 when people have provided a response, and a value of 0 when people have not. When the value of y is not observed, it is notated as $y = NULL$. These form triples $\boldsymbol{d} = (x, y, a) = \{(x_1, y_1, a_1), ..., (x_n, y_n, a_n)\}$. The basic inference problem, from one iteration to the next, is then, $p(h_t|\boldsymbol{d}) \propto p(d|h_{t-1}, h^*)p(h_{t-1})$,

$$p(h_t|\boldsymbol{d}) \propto p(y|x, a, h^*)p(a|y^*, x, h^*)p(x|h_{t-1})p(h_{t-1}) \tag{3.18}$$

where y^* represents the output that would be observed, if an action were taken. The main change is in people's choice of whether to respond, $p(a|y^*, x, h^*)$. A missing at random assumption implies that $p(a|y^*, x, h^*)$ does not depend on x, y, or h, thus $p(y|x, a, h^*) = p(a)$. If variables are missing due to a person's choice, the probability of a missing value almost certainly depends on x, y^* and/or h^*. We can formalize this choice using Luce choice [47], a special case of softmax [134][1],

$$p(a = 1|y^*, x, h^*) = \frac{U(a = 1|y^*, x, h^*)}{U(a = 0|y^*, x, h^*) + U(a = 1|y^*, x, h^*)} \tag{3.19}$$

where the choice of whether to act depends on the relative utility of acting as opposed to not acting. For example, if it is especially effortful to act, then people will be biased against acting. Alternatively, the utility of acting may depend on the value of y^*. For example, it may be that there is greater perceived utility in acting when the value of y^* is very low, as in the case of an angry customer or disappointed user.

In principle, one might think that this is related to the problem of dealing with missing data that is common in statistics [136]. Indeed, in our analyses, we showed one special case that reduces to the missing at random typically assumed in statistical applications [136]. However, the framework proposed here is in fact more general; it proposes a theory of why data are missing, and formalizes the problem as one of understanding human behavior [56, 137, 138].

[1] Both softmax and Luce choice have known issues for modeling human choice [47, 135]

3.5 Iterated Learning Mechanism

In order to study the interaction between humans and algorithms, we modify iterated learning so that it involves both a human and an algorithm instead of a succession of human agents. Figure 3.5 shows the flowchart of modified iterated learning involving both human and algorithm.

In this framework, first we randomly select some point for initialization, or it follows criterion such as biased initialization. Secondly the system employs a machine learning algorithm to training a model based on purpose, for example here we focus on a two-class recommendation problem. Third, the system applies the learned model to a candidate set of items or data, the system select the data points to present to the user based on different approaches. Following data selection, the system comes to the human interaction part where we need to simulate the probability of action taken by the user. If the user selects data X_i and labels it as y_i, our framework takes this new data and adds it to the training data, and finally proceeds to the next iteration.

Note that this framework has great flexibility in several parts. First, technically many machine learning algorithms are applicable. Second, the number of data items selected to show to the user can be varied according to interests. We choose one here as the number of items in order to be consistent with a human experiment, because a human will react only to one data point each time in our planned user experiment so that the system can investigate iterated machine learning model boundary shift in detail.

3.6 Evaluating the Effect of Iterated Algorithmic Bias on Learning Algorithms

3.6.1 Blind spot

The *blind spot* is defined as the set of data available to a relevance filter algorithm, for which the probability of being seen by the human interacting with the algorithm, that learned the hypothesis h is less than δ:

$$\mathbf{D}_{\delta}^{\mathbf{F}} = \{\mathbf{x} \in X \mid p_{seen}(\mathbf{x}|h) < \delta\} \tag{3.20}$$

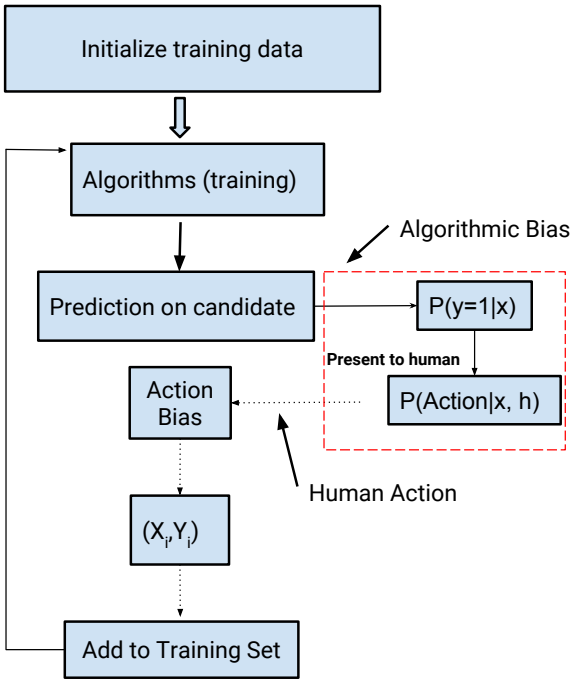


Figure 3.5: Flowchart of iterated learning involving both human and algorithm

In the real world, some data can be invisible to some users because of bias either from users or from the algorithm itself. Studying blind spots can enhance our understanding about the impact of algorithmic bias on humans. In addition, we define the *class-1-blind spot* or *relevant-item-blind spot* as the data in the blind spot, with true label $y = 1$

$$\mathbf{D}_{\delta}^{\mathbf{F}+} = \{\mathbf{x} \in \mathbf{D}_{\delta}^{\mathbf{F}} \quad \textbf{and} \quad \mathbf{y} = \mathbf{1})\} \tag{3.21}$$

Note that the blind spot in Eq. 3.20 is also called *all-classes-blind spot*.

3.6.2 Boundary shift

Boundary shifting indicates how different forms of iterated algorithmic bias affect the model h that is learned by an algorithm. It is defined as the number of points that are

predicted to be in class $y = 1$ given a learned model h:

$$b = \sum_{x \in X} p(y = 1|x, h) \tag{3.22}$$

Here b is the number of points that are predicted as class $y = 1$ given a learned model h.

3.6.3 Gini Coefficient

We also conduct a Gini coefficient analysis on how boundary shifts affect the inequality of predicted relevance for the test set. Let $p_i = p(y = 1|x_i, h)$. For a population with n values p_i, $i = 1$ to n, that are indexed in non-decreasing order ($p_{(i)} \leq p_{(i+1)}$). The Gini coefficient can be calculated as follows [139]:

$$G = (\frac{\sum_{i=1}^{n}(2i - n - 1)p_{(i)}}{n\sum_{i=1}^{n} p_{(i)}}) \tag{3.23}$$

The higher the Gini coefficient, the more unequal are the frequencies of the different labels. Given that the Gini coefficient measures the heterogeneity of the distribution of the relevance probabilities, it can be used to gauge the impact of different iterated algorithmic bias modes on the heterogeneity of the predicted probability in the relevant class during human-machine learning algorithm interaction.

3.7 Summary and Conclusions

In this chapter, We first compared our proposed iterated algorithmic bias and other types of bias in machine learning context. We then presented an iterated learning framework to simulate and study the learning mechanism with data dependency from one iteration to the next. Three approaches were presented to model iterated bias, namely, Filter-Bias, Active-Learning Bias, and Random selection.We also introduce several estimation metric to study the effect from iterated algorithmic bias.

CHAPTER 4

EXPERIMENTS ON ITERATED ALGORITHMIC BIAS IN HUMAN AND MACHINE LEARNING INTERACTION

Our preliminary results are based on both synthetic and real data. As stated in Chapter 3, we mainly focus on a two-class model of recommendation in order to perform our study. The classes are relevant/non-relevant, or like/dislike. In this situation, any classical supervised classification could be used in our model, such as Bayesian classifier [140], Logistic regression [141], or Support Vector Machine (SVM) [142]. For the purpose of easier interpretation and visualization of the boundary and to more easily integrate with the probabilistic framework in Section 3.3, we chose the Naive Bayes classifier.

4.1 Research Questions:

We first present several research questions we aim to address. The key issue is to show that information filtering may lead to systematic biases in the learned model, as captured by the classification boundary. Based on three metrics, we identify several research questions:

(RQ 1): How do different iterated algorithmic biases affect the behavior of models learned by a ML algorithm (with human action probability equal to 1)? We consider three aspects of a learned model to measure the outcome of algorithmic bias:

- **RQ 1.1) Boundary shift pre and post iteration (Eq. 3.22);**
- **RQ 1.2) Gini coefficient of predicted testing set labels (Eq. 5.10);**
- **RQ 1.3) Blind spot size pre and post iteration (Eq. 3.21).**

(RQ 2): How does class imbalanced initialization affect the model learned during iterative learning? We consider the same three aspects to measure the effects as in RQ 1.

(RQ 3): How does human action (whether to label data when requested to by the machine learning algorithm) affect the boundary shift? We consider the same three aspects to measure the effects as in RQ 1.

4.2 Data Sets

Our preliminary results are based on both synthetic and real data. As stated in Chapter 3, we mainly focus on a two-class model of recommendation in order to perform our study. The classes are relevant/non-relevant, or like/dislike. In this situation, any classical supervised classification could be used in our model, such as Bayesian classifier [140], Logistic regression [141], or Support Vector Machine (SVM) [142]. For the purpose of easier interpretation and visualization of the boundary and to more easily integrate with the probabilistic framework in Section 3.3, we chose the Naive Bayes classifier.

Synthetic Data: First, a 2D data set (see figure 4.1) was generated from two Gaussian distributions corresponding to classes $y \in \{0, 1\}$ for like (relevant) and dislike (non-relevant), respectively. Each class contains 1000 data points centered at $\{-2, 0\}$ and $\{2, 0\}$, with standard deviation $\sigma = 1$. The data set is then split into the following parts:

- Testing set: used as a global testing set (200 points from each class).
- Validation set: used for the blind spot analysis (200 points from each class).
- Initializing set: used to initialize the first boundary (we tested different initializations with class 1/class 0 ratios as follows: 10/100;100/100;100/10).
- Candidate set: used as query set of data which will be gradually added to the training set (points besides from the above three groups will be added to the candidate set).

The reason why we need the four subsets is that we are simulating a real scenario with interaction between human and algorithm. Part of this interaction will include picking

query data items and labeling them, thus augmenting the training set. Thus, to avoid depleting the testing set, we need to isolate these query items in the separate "candidate pool". A similar reason motivates the remaining separate subsets in order to keep their size constant throughout all the interactions of module learning.

We are also motivated to run experiments on high dimensional synthetic data set. We thus generate 3D, 4D and 10D synthetic data to study the effects of different iterated algorithmic bias in Section 4.4 chapter 4.

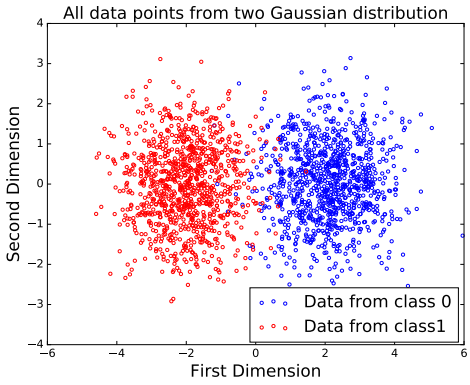


Figure 4.1: Original data with two classes

Real-life Data Set: In addition to synthetic data, we are motivated to use a real data set that expresses likes/dislikes that could be used as a two-category data set similar to the synthetic data set. Here we use the movielens dataset [143], which contains 100,004 ratings on 9125 movies. These ratings were made by 671 users between January 09, 1995 and October 16, 2016. The latest dataset was generated on October 17, 2016. All 671 users had rated at least 20 movies. The ratings range in [1, 5] and include missing values. We discretized the ratings into two labels: the range [1,3] was mapped to class 0, while (3,5] was mapped to class 1. The item content features were the movie genres. In order to perform similar experiments to the synthetic data simulations, we needed to focus on one user at a time. Thus we selected users who rated the highest number of movies and whose ratings are balanced between the two classes, analogously to our synthetic data. We first selected user $ID = 547$ to perform our study, then repeated the experiments on 7 more users. User 547 has rated 2312 movies throughout the 10 years. After removing some genres which appear

across the movies less than 10 times, we end up with 18 valid genres. We then perform Principal Component Analysis (PCA) to reduce the dimensionality with component cutoff as 0.90 [144], and ended up with 11 content features. One reason we perform PCA is to be more consistent with the Naive Bayesian assumption that all features are independent.

Methods: Each dataset contains two ground-truth categories of liked and disliked items. We wish to simulate the human-algorithm interaction at the heart of recommendation and information filtering. To do so, we consider three initialization possibilities: unbiased initialization in which examples are randomly selected from both categories in the same proportion; two types of biased sampling in which the relevant class (class 1) was oversampled by 10:1 or 1:10. Note that for the real-life data set, we only explore the unbiased initialization. We consider three forms of iterated algorithmic bias: random selection, active-bias which attempts to learn the true boundary between the two categories, and filter (or recommendation) bias which attempts to recommend only preferred items (see Section 3.3). We simulate different types of responses by the user as action probabilities that vary from labeling each item as it is recommended (human action probability of 1), to two cases where the user labels only some of the items provided by the algorithm (human action probabilities of 0.5 and 0.01). Note that absent any additional information, we assume action probabilities to be 1 for the real-life data set. We then simulate runs of 200 iterations where a single iteration consists of the algorithm providing a recommendation, the user labeling (or not) the recommendation, and the algorithm updating its model of the user's preferences. Each combination of parameters yields a data set that simulates the outcome of the human and algorithm interacting. We simulate this whole process 40 times independently, which generates the data that we will use to investigate the research questions listed in Section 4.1.

In this section, we present results using both synthetic and real-life data. We first present results from the 2D synthetic data described in Section 4.4. We also explore high dimensional synthetic data in Section 4.4. The real-life data set experiments are presented in Section 4.5. Finally, we summarize the conclusions from all our synthetic data experiments

in Table 4.27, and from the real-life data experiments in Table 4.28.

4.3 Experiments on 2D Synthetic Data

4.3.1 RQ 1: How does iterated algorithmic bias affect the learned categories?

To answer this question, we control for human action bias by assuming the data are labeled in each iteration, i.e. the probability of action is 1, $p_{action} = 1$. We adopt four different approaches to investigating this question. First, we will compare the inferred boundaries after interaction to the ground truth boundaries. Second, we will focus on the effects of iteration alone by analyzing the boundary before interaction and after. Third, We use the Gini coefficient to measure the heterogeneity or inequality of the predicted label distribution in the testing set. Fourth, we investigate the size of the blind spot induced by each of the iterated algorithmic bias modes. Together, these will describe the outcomes of algorithmic bias, in terms of how it interacts with initialization, and the consequences of algorithmic bias in terms of the induced blind spot.

RQ 1.1: Do different forms of iterated algorithmic bias have different effects on the boundary shift?

To answer this question, we control for human action bias by assuming the data are labeled in each iteration i.e. the probability of action is 1. And the initialization is balanced, i.e the ratio=1:1. As shown in Eq. 3.5, we here assume that $q(x)$ is identical for all data points, thus we can ignore the second part of the equation, i.e. the probability of being seen is only dependent on the predicted probability of candidate points. Note that we could get some prior probability of X_i, in which case we could add this parameter to our framework. Here, we assume them to be the same, hence we set $\epsilon = 0$.

We wish to quantify differences in the boundary between the categories as a function of the different algorithm biases. To do so, we generate predictions for each test point in the

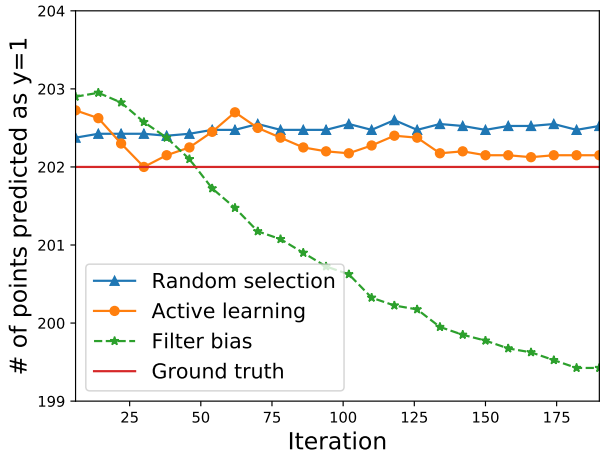


Figure 4.2: Boundary shift (Eq. 3.22) based on the three iterated algorithmic bias forms. The y-axis is the number of testing points which are predicted to be in class y=1. This figure shows that random selection and active learning bias do not have significant effect with iteration goes on, and converge to the ground truth boundary. Filter bias, on the other hand, results in decreasing numbers of points predicted in the target category class 1, consistent with an overly restrictive category boundary. The ground truth shows the actual proportion of points with class y=1 in the testing set.

test set by labeling each point based on the category that assigns it highest probability. We investigate the proportion of test points with the relevant label $y = 1$ at two time points: prior to human-algorithm interactions (immediately after initialization), and after human algorithm interactions.

We run experiments with each of the three forms of algorithm bias, and compare their effect on boundary shift. We also report the effect size based on *Cohen d* algorithm [145]. In this experiment, the effect size (ES) is calculated by $ES = (Boundary_{t=0} - Boundary_{t=200})/standard.dev$, here $standard.dev$ is the standard deviation of the combined samples. We will use the same strategy to calculate the effect size in the rest of this paper. The results indicate significant differences for the filter bias condition ($p < .001$ by Mann-Whitney test or t-test, effect size = 1.96). In contrast, neither the Active Learning, nor the Random conditions resulted in statistically significant differences ($p = .15$ and .77 by Mann-Whitney test, or $p = .84$ and 1.0 by t-test; effective sizes .03 and 0.0, respectively).

To illustrate this effect, we plot the number of points assigned to the target category

versus ground-truth for each iterations. Figure 4.2 shows that random selection and active learning bias do not have significant effect with iteration goes on. Filter bias, on the other hand, results in decreasing numbers of points predicted in the target category class 1, consistent with an overly restrictive category boundary.

RQ 1.2: Do different iterated algorithmic bias modes lead to different trends in the inequality of predicted relevance throughout the iterative learning, given the same initialization?

In order to answer this question, we run experiments with different forms of iterated algorithmic bias, and record the Gini coefficient when a new model is learned and applied to the testing set during the iterations. We first run the Shapiro-Wilk normality test with all groups [146]. The p-value for filter bias, active learning bias and random selection are 0.91, 0.63 and 0.99 respectively. Therefore, we perform a one-way ANOVA tests [147].

Although the absolute difference between the first iteration and the last iteration is small (see Figure 4.3), a one-way ANOVA test across these three iterated algorithmic bias forms shows that the Gini index values are significantly different. The p-value from the ANOVA test is close to 0.000 (<0.05), which indicates that the three iterated algorithmic bias forms have different effects on the Gini coefficient.

Interpretation of this result: Given that the Gini coefficient measures the inequality or heterogeneity of the distribution of the relevance probabilities, this simulated experiment shows the different impact of different iterated algorithmic bias forms on the heterogeneity of the predicted probability to be in the relevant class within human machine learning algorithm interaction. Despite the small effect, the iterated algorithmic bias forms affect this distribution in different ways, and iterated filter bias causes the largest heterogeneity level as can be seen in Figure 4.3.

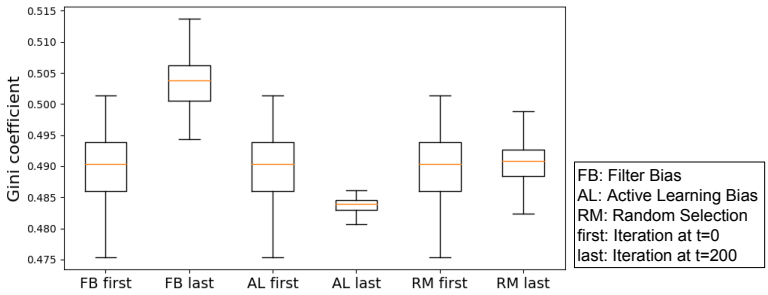


Figure 4.3: Box-plot of the Gini coefficient resulting from three forms of iterated algorithmic bias. An ANOVA test across these three iterated algorithmic bias forms shows that the Gini index values are significantly different. The p-value from the ANOVA test is close to 0.000 (< 0.05), which indicates that the three iterated algorithmic bias forms have different effects on the Gini coefficient. Here, FB, AL and RM are the abbreviations of filter bias, active learning bias and random selection, respectively. The 'first' indicates the beginning of iteration (i.e., t=0), while the 'last' means the end of iteration (i.e., t=200). Note that we use these abbreviations in the rest of our paper.

RQ 1.3: Does iterated algorithmic bias affect the size of the class-1-blind spot and the all-classes-blind spot, i.e. is the initial size of the blind spot $\mathbf{D}_{\delta}^{\mathbf{F}}$ significantly different compared to its size in the final iteration?

The blind spot represents the set of items that are much less likely to be shown to the user. Therefore this research question studies the significant impact of an extreme filtering on the number of items that can be seen or discovered by the user, within human - algorithm interaction. If the size of the blind spot is higher, then iterated algorithmic bias results in hiding items from the user. In the case of the blind spot from class 1, this means that even relevant items are affected.

Recall from section 3.6 that some of the validation set data points have high probability to be seen, while others have low probability to be seen, the latter make up what we refer to as the blind spot. We study how iterated algorithmic bias affects the size of the blind spot. Here, δ is the threshold on the probability of being seen for an item to be considered in the blind spot. Recall that the blind spot items from class $y = 1$ are called *relevant item blind spot* or *class-1-blind spot* and the items from both classes are called

TABLE 4.1: Results of the Mann-Whitney U test and t-test comparing the size of the class-1-blind spot for the three forms of iterated algorithmic bias. The effect size is $(BlindSpot|_{t=0} - BlindSpot|_{t=200})/standard.dev$. Bold means significance at $p < 0.05$. The negative effect size shows that filter bias increases the class-1-blind spot size. For active learning bias, the p-value indicates the significance, however the effect size is small. Random selection has no significant effect.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**2.4e-10**	**0.03**	0.06
t-test p-value	**2.2e-10**	**0.03**	0.06
effect size	**-1.22**	**-0.47**	-0.4

all-classes-blind spot.

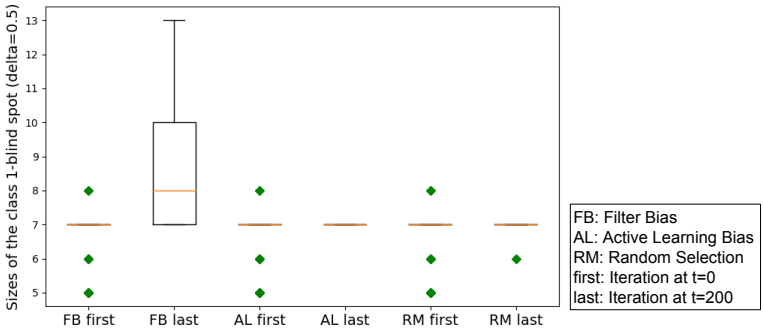


Figure 4.4: Box-plot of the size of the class-1-blind spot for all three iterated algorithmic bias forms. In this figure, the x-axis is the index of the three forms of iterated algorithms biases. As shown in this box-plot, the initial class-1-blind spot is centered at 7. This is because the 200 randomly selected initial points from both classes force the boundary to be similar regardless of the randomization.

We run experiments with $\delta = 0.5$ for the class-1-blind spot, and record the size of the class-1-blind spots with three different iterated algorithmic bias forms. Here, we aim to check the effect of each iterated algorithmic bias form. Filter bias has significant effects on the class-1-blind spot, while random selection and active learning do not have a significant effect on the class-1-blind spot size (see Figure 4.4). As shown in Figure 4.4), around $8/400 = 2\%$ of the relevant points in testing set are hidden. The negative effect from iterated filter bias implies a large increase in the class 1 blind spot size, effectively hiding a significant number of 'relevant' items. Table 4.1 summarizes the result of the statistical analysis. Note that the effect size is calculated by $ES = (BlindSpot|_{t=0} - BlindSpot|_{t=200})/standard.dev$.

TABLE 4.2: Results of the Mann-Whitney U test and t-test comparing the size of the all-classes-blind spot for the three forms of iterated algorithmic bias. The effect size is conducted as $(BlindSpot|_{t=0} - BlindSpot|_{t=200})/standard.dev$. The negative effect size shows that filter bias increases the class-1-blind spot size. On the other hand, both active learning and random selection have no significant effect.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**5.9e-12**	0.5	0.47
t-test p-value	**1.4e-19**	0.18	0.13
effect size	**-1.44**	-0.3	-0.29

We perform a statistical test on the all-classes-blind spot with $\delta = 0.5$. We first run the Shapiro-Wilk normality test. The p-values for the first and last iterations are respectively, 2.74e-7 and 0.037 for filter bias; 2.74e-7 and 1 for active learning bias; and 2.74e-7 and 2.9e-11 for random selection. Therefore, we perform a non-parametric statistical test on the pairs of data using the Mann-Whitney U test [148]. The p-value from the Mann-Whitney U test is close to 0.000 for filter bias. The negative effect from iterated filter bias implies a large increase in the all-class blind spot size, effectively hiding a significant number of 'relevant' and irrelevant items. For AL, the effect size has no significant effect in hidden items (both relevant and non-relevant) based on the Mann-Whitney U test. Random selection results in no significant effect on the blind spot size as well. Similar results can be found in the all-classes blind spot size experiment results (see Table 4.2).

Interpretation of this result: Given that the blind spot represents the items that are much less likely to be shown to the user, this simulated experiment studies the significant impact of an extreme filtering on the number of items that can be seen or discovered by the user, within human-machine interaction. Iterated filter bias effectively hides a significant number of 'relevant' items that the user misses out on compared to AL. AL has no significant impact on the relevant blind spot, but increase the all-class blind spot to certain degree. Random selection has no such effect.

TABLE 4.3

Prediction accuracy of different imbalanced initializations ratio (class 1: class 0) on the testing set and ground-truth boundary

	ground truth	ratio=1:1	ratio=10:1	ratio=1:10
accuracy	0.985	0.982	0.94	0.95

4.3.2 RQ 2: Does class imbalanced initialization affect the boundary learned during iterative learning?

Online systems which have a very wide set of options and where users tend to provide initial ratings for items that they like or see, do suffer from initial class imbalance. Class imbalance can also emerge from the algorithm intentionally asking users to rate only the most popular items, a common strategy used to collect initial ratings for new users. It is also important to notice that in different domains, initialization can have different imbalance patterns. For example in the popular Movielens-100k dataset [143], around 65% of users have rated more items with ratings higher than their own average rating. A similar phenomenon can be observed in other data sets such as Movielens-1M [143], Movelens-10M [143], Netflix Prize challenge dataset [149], and Book Crossings data set [150]. There are also users who have rated more items with lower ratings.

We want to measure how imbalanced initialization affects the algorithm's performance. We set up three different class imbalance initialization ratios (class0 : class1): 1:10, 1:1 and 10:1. Then we compare the learned boundaries with those three ratios. As shown in Figure 4.5, highly imbalanced class initialization leads to a bigger difference between the learned boundary and the ground-truth boundary. The ground-truth boundary is obtained from the Gaussian distributions that were used to generate the data points in Section 4.2. We also quantify the difference between the learned and true boundaries by measuring how initial boundaries predict the labels on the testing set. The accuracy of the ground-truth boundary and three imbalanced initial boundaries are recorded in Table 4.3. We can see that the increase in initialization imbalance ratio leads to lower accuracy on the testing set. Note that accuracies are averaged from 10 independent runs.

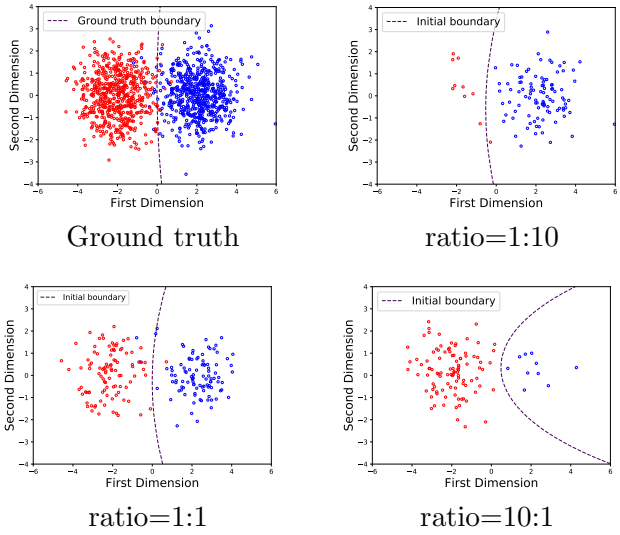


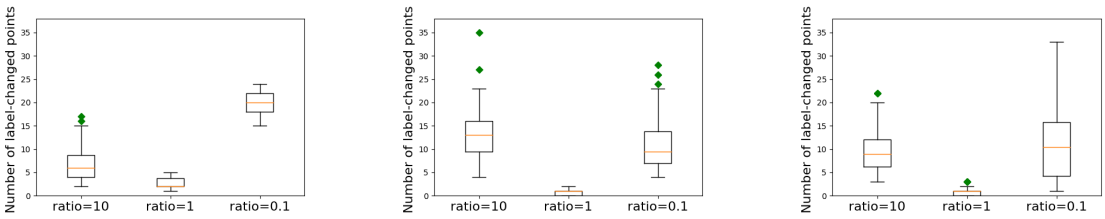
Figure 4.5: Different initialization imbalance ratios affect the starting model boundary ax expected

We wish to understand how imbalanced initialization affects the boundary shifting and the blind spot size during the interactive learning process. To answer this question, we run experiments 40 times with the different class imbalanced initialization ratios and record the number of points which cross the boundary during iterative learning as well as the blind spot size. We will consider three imbalanced initialization (class 1: class 0) ratios, namely 10:1, 1:1 and 1:10.

RQ 2.1: Does class imbalanced initialization affect the boundary learned during iterative learning given a fixed iterated algorithmic bias mode?

In order to answer this question, we record the number of points whose labels are different between the first iteration and last iterations, with imbalanced initialization ratio set to 10:1, 1:1 and 1:10. We first perform the Shapiro-Wilk normality test with all groups [146]. The p-value are 0.002, 0.001 and 0.06 for filter bias with four ratios; 0.01, 0.0003 and 0.001 for active learning bias; and 0.01, 0.0003 and 0.07 for random selection. Therefore, we perform a non-parametric statistical test using the Kruskal-wallis test [34] on each form of algorithmic bias. Figure 4.6 show the trends of label changes. The Number of label-changed points with ratio=10:1 is higher than 2 for ratio=1:1 in the iterated filter bias

mode, and a similar result can be seen with iterated active learning bias and iterated random selection (see Figure 4.6). The p-values from the Kruskal-wallis test for all three forms of iterated algorithmic bias are 5.5e-19 (filter bias), 3.27e-17 (active learning bias) and 1.34e-15 (random selection), close to 0.00, which indicates that the class imbalance initialization affects the boundary shift for all three forms of bias. Also, the higher the class imbalanced initialization ratio, the more boundary shifting, as shown in Figure 4.6. On the other hand, the boundary shift of filter bias has a big difference when the ratio is 1:10, indicating that filter bias has a dramatic impact on the boundary shift when the imbalanced initialization ratio is high and more points are from the irrelevant class.



(a) Filter Bias (b) Active Learning (c) Random Selection

Figure 4.6: The box-plots showing the distribution of the number of points that moved across the boundary in the first and last iterations of the iterated learning for three iterated algorithmic biases with different class imbalanced initialization ratios. Because the number of points that move across the boundary indicates the intensity of the boundary shift, this shows that for all three iterated algorithmic bias modes, a high imbalanced initialization results in a higher boundary shift compared to a balanced initialization.

Interpretation of this result: Given that the number of label-changed points represents the number of items that move across the relevance boundary learned by the machine learning algorithm, this simulated experiment studies the impact of an initial class imbalance (a ratio of 10:10 or 1:10 versus a ratio of 1:1) when an extreme filtering strategy is used within human machine interaction.

In all cases and regardless of whether extreme filtering, AL, or random selection is used to collect feedback from the user, the initial class imbalance has a significant impact on the shift in the learned model's boundary between relevant and non-relevant items. This test confirms that a more drastic initial class imbalance in the training data has a significant

impact on the resulting boundary, and as a result on the judgment of items to be relevant or not by the learned model. Online systems which have a very wide set of options and where users tend to provide initial ratings for items that they like or see, do suffer from initial class imbalance. Class imbalance can also emerge from the algorithm intentionally asking users to rate only the most popular items, a common strategy used to collect initial ratings for new users.

RQ 2.2: Does class imbalanced initialization affect the blind spot size during iterative learning given a fixed iterated algorithmic bias mode?

We run 40 experiments with the different initialization ratios 10:1 and 1:10 and record both class-1-blind spot and all-classes-blind spot size. We first run the Shapiro-Wilk normality test on the class-1-blind spot size records. The p-values for imbalance ratio 10:1, for the first and last iterations were 5.5e-6 and 0.0001 for filter bias; 5.5e-6 and 1 for active learning bias; and 5.5e-6 and 1.0e-8 for random selection. Therefore, we perform a non-parametric statistical test on the pairs of data using the Mann-Whitney U test [148]. The p-value from the Mann-Whitney U test was close to 0 for all three different iterated bias modes. The negative effect implies a large increase in the all-class blind spot size, implying that a 10:1 class imbalanced initialization results in effectively hiding a significant number of 'relevant' and irrelevant items (see Table 4.5). We also perform a statistical test on all-classes-blind spot size with 10:1 imbalanced initialization ratio (see table 4.5). The p-value for all three iterated algorithmic biases are close to 0, which indicates that all algorithmic bias modes have significant impact on the blind spot size, when we have a highly imbalanced initialization. We also report the blind spot analysis with initialization ratio 1:10 in table 4.1 and table 4.2. A statistical test on the all-classes-blind spot size (see table 4.5) yielded a p-value close to 0 for all three iterated algorithmic biases, which indicates that they all have significant impact on the blind spot size, when starting with a highly imbalanced initialization.

A Shapiro-Wilk normality test on the class-1-blind spot with ratio 1:10 yielded the

TABLE 4.4: Results of the Mann-Whitney U test and t-test comparing the size of the class-1-blind spot for the three forms of iterated algorithmic bias with imbalanced (class 1:class 0) initialization ratio 10:1 and 1:10. The effect size is $(BlindSpot|_{t=0} - BlindSpot|_{t=200})/standard.dev$. The negative effect size with ratio=10:1 shows that all three iterated algorithm bias modes increase the class-1-blind spot size. On the other hand, filter bias increases the class-1-blind spot size with ratio 1:10, and both active learning and random selection decrease the class-1-blind spot size.

	Measurement	Filter Bias	Active Learning	Random Selection
ratio=10:1	Mann-Whitney test p-value	**4.3e-9**	**5.6e-17**	**1.7e-15**
	t-test p-value	**1.0e-14**	**8.4e-28**	**5.0e-24**
	effect size	**-1.33**	**-1.91**	**-1.86**
ratio=1:10	Mann-Whitney test p-value	**2.7e-13**	**3.6e-16**	**1.1e-9**
	t-test p-value	**2.2e-30**	**9e-10**	**1.1e-15**
	effect size	**-1.6**	**1.34**	**1.33**

TABLE 4.5: Results of the Mann-Whitney U test and t-test comparing the size of the all-classes-blind spot for the three forms of iterated algorithmic bias with imbalanced (class 1:class 0) initialization ratio 10:1 and 1:10. The results are similar to the class-1-blind spot size analysis (see figure 4.4).

	Measurement	Filter Bias	Active Learning	Random Selection
ratio=10:1	Mann-Whitney test p-value	**1.0e-9**	**3.5e-15**	**5.3e-15**
	t-test p-value	**6.1e-21**	**1.0e-18**	**2.0e-17**
	effect size	**-1.31**	**-1.75**	**-1.75**
ratio=1:10	Mann-Whitney test p-value	**2.7e-13**	**9.5e-17**	**4.5e-15**
	t-test p-value	**1.6e-30**	**4.8e-13**	**4.7e-12**
	effect size	**-1.6**	**1.53**	**1.48**

following p-values for the first and last iterations respectively for each type of algorithmic bias: 0.0003 and 0.0003 for filter bias; 0.00031 and 1 for active learning bias; and 0.0031 and 3e-13 for random selection. Therefore, we perform a non-parametric statistical test on the pairs of data using the Mann-Whitney U test [148]. The p-value from the Mann-Whitney U test is close to 0 for all three different iterated bias modes. The negative effect implies a large increase in the all-classes-blind spot size for filter bias mode, effectively hiding a significant number of 'relevant' and irrelevant items (see Table 4.4). Meanwhile, active learning and random selection lead to a significant decrease in blind spot size.

In addition, we compare the blind spot size in the last iteration of model learning across different algorithmic bias modes themselves given a fixed imbalanced initialization

ratio. Note here that all three modes have the same initialization points (same initial boundary in each run). Table 4.6 shows the paired t-test results among the three iterated algorithmic bias modes with imbalanced initialization ratio 10:1. The effect size is calculated as $ES = (BlindSpot|_{row} - BlindSpot|_{column})/standard.dev$. For example, we compare the bias type of row 1 to bias type of column 4 (i.e., Filter bias vs. active learning). We can see that filter bias has lower impact on the blind spot than active learning and random selection when the system is initialized with high class imbalance. Active learning also has higher impact on the all-classes-blind spot. Table 4.6 also shows the paired t-test result among the three iterated algorithmic bias modes with imbalanced initialization ratio 1:10, a similar result to the one obtained when the ratio is 1:1. We can therefore see that filter bias has a higher impact on the blind spot size than active learning and random selection, while active learning has higher impact on the blind spot than random selection.

We also compare the impact on the blind spot size with different initialization ratios (balanced vs. highly imbalanced initialization) given a fixed iterated algorithmic bias mode. The effect size is calculated using $ES = (BlindSpot|_{ratio=1:1} - BlindSpot|_{ratio=10:1})/standard.dev$ at time $t = 200$. Table 4.7 shows the p-value resulting from comparing the blind spot size between ratio 1:1 and ratio 10:1 for each of the iterated algorithmic bias modes. Table 4.7 shows that imbalanced initialization has lower impact on filter bias than an active learning and random selection, which indicates that balanced initialization with filter bias worsens the ability of humans to discover new items. Table 4.8 shows the all-classes-blind spot size difference between (class 1: class 0) ratios 1:1 and 1:10. Here $ES = (blindSpot|_{ratio=1:1} - BlindSpot|_{ratio=1:10})/standard.dev$ at $t = 200$. We conclude that filter bias and random selection lead to increasing the blind spot size when the initial imbalance ratio is 1:10 compared with balanced initialization, which indicates that relevant items will be more likely to be hidden in case of highly imbalanced initial labeled data with more points from the irrelevant class (class y=0). On the other hand, active learning bias has similar impact with ratios 1:1 and 1:10.

TABLE 4.6: Results of the t-test comparing the size of the all-classes-blind spot across the three forms of iterated algorithmic bias with imbalanced (class 1:class 0) initialization ratio 10:1 and 1:10 at iteration t=200. The p-values from both ratio=1:10 and ratio=10:1 indicate the significant difference between different algorithmic bias modes. The effect size indicates that ratio=1:10 and ratio=10:1 lead to opposite trends on the comparison.

	Bias type	Filter Bias	Active Learning	Random Selection
ratio=10:1	Filter Bias	–	**p=1.1e-10** (effect size=-1.07)	**p=6e-5** (effect size=-0.9)
	Active Learning	–	–	**p=8e-8** (effect size=1)
ratio=1:10	Filter Bias	–	**p=9.5e-17** (effect size=1.9)	**p=3.2e-15** (effect size=1.89)
	Active Learning	–	–	**p=1.96e-10** (effect size=-1.2)

TABLE 4.7: Results of the Mann-Whitney U test and the t-test comparing the size of the all-classes-blind spot for the three forms of iterated algorithmic bias based on different initialization balance (between ratio=1:1 and ratio=10:1). Here effect size is $ES = (blindSpot|_{ratio=1:1} - BlindSpot|_{ratio=10:1})/standard.dev$ at $t = 200$. The positive effect sizes shows that Both filter bias and random selection have higher impact when the imbalanced initialization ratio is 1:1. Active learning is insensitive to the initialization ratio.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**1.0e-14**	1.0	**0.0001**
t-test p-value	**5e-12**	1.0	**0.0002**
effect size	**1.48**	0.0	**0.8**

TABLE 4.8: Results of the Mann-Whitney U test and the t-test comparing the size of the all-classes-blind spot for the three forms of iterated algorithmic bias based on different initialization (between ratio=1:1 and ratio=1:10). Here effect size is $ES = (blindSpot|_{ratio=1:1} - BlindSpot|_{ratio=1:10})/standard.dev$ at $t = 200$. The negative effect size shows that Both filter bias and random selection have higher impact when the imbalanced initialization ratio is 1:10. Active learning is insensitive to the initialization ratio.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**6.5e-15**	1.0	**9e-9**
t-test p-value	**5.2e-24**	1.0	**1.5e-16**
effect size	**-1.85**	0.0	**-1.1**

RQ 2.3: Does class imbalanced initialization affect the inequality of prediction during iterative learning given a fixed iterated algorithmic bias mode?

In order to answer this question, we perform experiments with different forms of iterated algorithmic bias, and record the Gini coefficient when a new model is learned and applied to the testing set during the iterations.

We first aim to check the effect across different forms of iterated algorithmic bias with class imbalance ratio 10:1 (see Figure 4.7). We first run the Shapiro-Wilk normality test with all groups [146]. The p-value for filter bias, active learning bias and random selection are 2.3e-10, 5.7e-6 and 1.5e-9, respectively. Therefore, we perform a non-parametric statistical test with the Kruskal-wallis test [34]. The Kruskal-wallis test across these three iterated algorithmic bias forms shows that the Gini index values are significantly different. The p-value from the Kruskal-wallis test is 3.6e-6 (<0.05), which indicates that the three iterated algorithmic bias forms have different effects on the Gini coefficient. We then check the effect across different forms of iterated algorithmic bias with ratio 1:10, the p-value from the Kruskal-wallis test is 1.0e-21 (<0.05).

We also run the statistical analysis for each iterated algorithmic bias based on the Gini coefficient between the first and last iteration during iterative learning. Table 4.9 shows that the filter bias mode will increase the inequality of predictions from the learned model in both cases, class imbalance ratio 10:1 and ratio 1:10. However, active learning and random selection have totally different impacts on the inequality. When the initial data is oversampled from class 1, all three bias modes result in increased inequality of prediction. On the other hand, when the system is oversampled from class 0, active learning and random selection bias modes help learning the correct boundary.

It is also interesting to compare how different iterated algorithmic bias modes affect the inequality given the same initialization with a fixed ratio. Table 4.10 shows that the there is no significant difference between filter bias and active learning bias modes with ratio 10:1. But, when the ratio is 1:10, filter bias results in significantly higher inequality compared with active learning bias and random selection, which is consistent with figure

TABLE 4.9: Results of the Mann-Whitney U test and the t-test comparing the inequality of predictions for the three forms of iterated algorithmic bias based on the first and last iteration with imbalanced initialization ratio 10:1 and 1:10. The effect size is $(Gini|_{t=0} - Gini|_{t=200})/standard.dev$. The negative effect size with ratio=10:1 shows that all three iterated algorithm bias modes increase the inequality. On the other hand, filter bias increases the inequality with ratio 1:10 and both active learning and random selection decrease the inequality.

	Measurement	Filter Bias	Active Learning	Random Selection
ratio=10:1	Mann-Whitney test p-value	**3.8e-12**	**5.7e-14**	**1.5e-14**
	t-test p-value	**1e-27**	**6.9e-14**	**7.3e-18**
	effect size	**-1.49**	**-1.58**	**-1.64**
ratio=1:10	Mann-Whitney test p-value	**3.2e-13**	**7.2e-15**	**8e-15**
	t-test p-value	**3.3e-34**	**1.9e-18**	**6.9e-13**
	effect size	**-1.68**	**1.74**	**1.5**

TABLE 4.10: Results of the t-test comparing the Gini coefficient across the three forms of iterated algorithmic bias with imbalanced initialization ratio 1:10 and 10:1 at time t=200. The effect size is $(Gini|_{row} - Gini|_{column})/standard.dev$ at t=200. Regarding the inequality, filter bias has similar impact with active learning and random selection with the imbalance initialization ratio is 10:1, while active learning bias mode leads to less inequality than random selection. When the initialization ratio is 1:10, filter bias lead to more inequality than both active learning bias and random selection. Active learning bias mode leads to less inequality than random selection.

	Bias type	Filter Bias	Active Learning	Random Selection
ratio=10:1	Filter Bias	–	p=0.51 (effect size=0.16)	p=0.11 (effect size=-0.3)
	Active Learning	–	–	**p=5.4e-7** (effect size=-1)
ratio=1:10	Filter Bias	–	**p=3.2e-31** (effect size=1.94)	**p=7.2e-29** (effect size=1.91)
	Active Learning	–	–	**p=1.3e-22** (effect size=-1.8)

4.7.

We also compare the Gini coefficient score between balanced class initialization (ratio 1:1) and highly imbalanced initialization (ratio 10:1). Table 4.11 shows the statistical result. As seen in table 4.11, a balanced initialization has higher impact on the Gini coefficient for filter bias and random selection. Table 4.12 shows the statistical test results comparing ratio 1:1 and ratio 1:10. We can see that balanced initialization leads to less inequality than imbalanced initialization (ratio 1:10). A possible reason is that the boundary moves

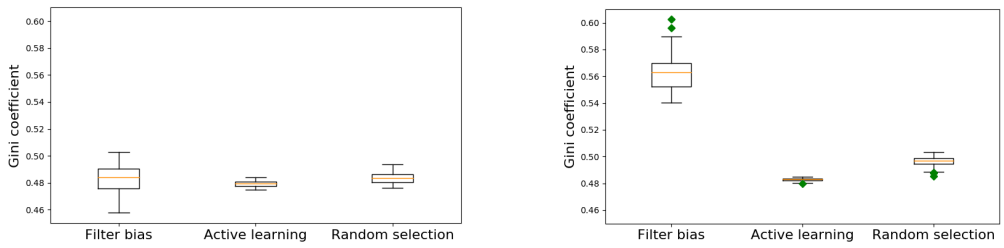
TABLE 4.11: Results of the Mann-Whitney U test and the t-test comparing Gini coefficients for the computed three forms of iterated algorithmic bias based on different initialization (between ratio 1:1 and ratio 10:1). Here effect size is $ES = (Gini|_{ratio=1:1} - Gini|_{ratio=10:1})/standard.dev$ at $t = 200$. The positive effect size shows that a balanced initialization ratio results in inequality lower for for all three iterated algorithmic bias modes.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**7.1e-14**	**9.6e-12**	**6.3e-9**
t-test p-value	**1e-12**	**1e-11**	**6.3e-9**
effect size	**1.49**	**1.50**	**1.28**

TABLE 4.12: Results of the Mann-Whitney U test and the t-test comparing Gini coefficients for the three forms of iterated algorithmic bias based on different initialization (between ratio 1:1 and ratio 1:10). Here effect size is $ES = (Gini|_{ratio=1:1} - Gini|_{ratio=1:10})/standard.dev$ at $t = 200$. The negative effect sizes shows that higher initial imbalance results in more prediction inequality for both filter bias and random selection modes.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**7.2e-15**	**0.0007**	**1.9e-7**
t-test p-value	**3.5e-26**	**0.001**	**1.94e-7**
effect size	**-1.88**	**0.73**	**-1.1**

towards the data from class 1 and away from class 0 with ratio 1:10, instead of being close to the middle with ratio 1:1.



(a) ratio=10:1 (b) ratio=1:10

Figure 4.7: Box-plot of the Gini coefficient resulting from three forms of iterated algorithmic bias with imbalanced initialization ratio set to 10:1 and 1:10. A Kruskal-wallis test across these three iterated algorithmic bias forms shows that the Gini index values are significantly different with both cases. The p-value from the Kruskal-wallis test are 3.6e-6 and 1.0e-10 for ratio 10:1 and ratio 1:10, which indicates that the three iterated algorithmic bias forms have different effects on the Gini coefficient from both cases.

4.3.3 RQ 3: Does human action bias affect the boundary?

In order to test how the human reaction affects the boundary shift, we set $p_{action} = 1.0$, 0.5 and 0.01, and record the results with different iterated algorithmic bias modes.

RQ 3.1: Does human action affect the boundary shift during iterative learning given a fixed iterated algorithmic bias mode?

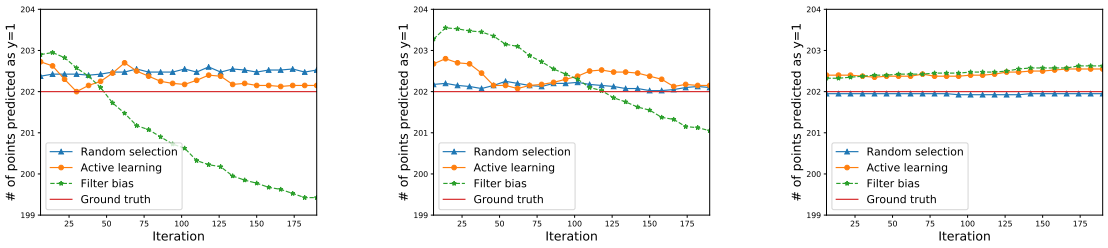
We first want to compare the shift in the boundary induced by the different algorithmic bias forms, alone. We do so by analyzing the change in the boundary, i.e., the number of points in the test set which are predicted to be in class y=1. We perform the Mann-Whitney U statistical test to see whether the different human action probabilities affect the boundary shift of the learned model for each iterated algorithmic bias and three possible human action probability levels. We also record results from the t-test. Effect size is also recorded between any pair of sets of boundary shift from the first iteration and last iteration. Table 4.13 shows that p_{action} affects the boundary shift more with iterated filter bias than with the other two forms of bias, which support the same conclusion as the previous experiment. The Mann-Whitney U test results agree with the t-test. Thus we conclude that human action affects the boundary shift: the more frequent human action is, the more significant is the effect on the boundary shift for only the iterated filter algorithmic bias. The other bias modes are not affected by different human action probabilities. Note that the effect size is calculated by the difference between the boundary of $t = 0$ and $t = 200$ divided by the standard deviation.

In addition, it is interesting to compare the final learned boundaries from the three iterated algorithmic bias forms with the ground truth. Therefore, we follow the same procedure as in Section 4.3.2, comparing the learned boundary during the iterations with the ground truth boundary. Figure 4.8 shows the results for different iterated algorithmic bias modes with different human action probability. We can see that for both random selection and active learning bias, the number of points predicted as relevant has no significant change

TABLE 4.13: Results of the Mann-Whitney U test and t-test for the boundary shift of the computed three forms of iterated algorithmic bias based on the different probability levels of human action. Here, effect size is $ES = (Boundary|_{t=0} - Boundary|_{t=200})/standard.dev$. Bold means the significance at $p < 0.05$. The more the human reacts to the system, the larger is the boundary shift.

	Measurement	Filter Bias	Active Learning	Random
$p_{action} = 1$	Mann-Whitney test p-value	**0.4e-11**	**0.9e-5**	0.486
	t-test p-value	**7.1e-14**	**0.002**	0.52
	effect size	**3.087**	**0.24**	-0.15
$p_{action} = 0.5$	Mann-Whitney test p-value	**1.8e-6**	0.283	0.229
	t-test p-value	**1.0e-5**	0.08	0.75
	effect size	**0.87**	0.19	0.08
$p_{action} = 0.01$	Mann-Whitney test p-value	0.259	0.336	0.42
	t-test p-value	0.38	0.6	0.9
	effect size	-0.27	0.14	-0.08

regardless the probability of human action. On the other hand, filter bias tends to have a lower proportion of points predicted to be relevant with higher human action probability. With $p_{action} = 0.01$, there are no obvious trends for all three algorithmic bias forms.



(a) p_{action}=1.0 (b) p_{action}=0.5 (c) p_{action}=0.01

Figure 4.8: Effect of Human action on the boundary learned during the iterations. The y-axis is the number of testing points which are predicted to be relevant. We can see that for both random selection and active learning bias, the number of points predicted as relevant has no significant change regardless the probability of human action and the number of points predicted as relevant converges to the ground truth boundary when there is a high probability of human action. On the other hand, filter bias tends to have a lower proportion of points predicted to be relevant with higher human action probability. With $p_{action} = 0.01$, there are no obvious trends for all three algorithmic bias forms.

TABLE 4.14: Results of the Mann-Whitney U test and t-test for the class-1-blind spot of the computed three forms of iterated algorithmic bias based on the different probability levels of human action. Here, effect size is $ES = (BlindSpot|_{t=0} - BlindSpot|_{t=200})/standard.dev$. Bold means the significance at $p < 0.05$. The more the human reacts to the system, the bigger is the class-1-blind spot size.

	Measurement	Filter Bias	Active Learning	Random
$p_{action} = 1$	Mann-Whitney test p-value	**2.4e-10**	**0.03**	0.06
	t-test p-value	**2.2e-10**	**0.03**	0.06
	effect size	**-1.22**	**-0.47**	-0.4
$p_{action} = 0.5$	Mann-Whitney test p-value	**0.0002**	0.49	0.50
	t-test p-value	**7e-5**	0.66	0.67
	effect size	**-0.8**	-0.1	-0.1
$p_{action} = 0.01$	Mann-Whitney test p-value	0.16	0.5	1.0
	t-test p-value	0.16	1	0.5
	effect size	0.02	0.0	0.0

RQ 3.2: Does human action affect the class-1-blind spot size during iterative learning given a fixed iterated algorithmic bias mode?

We want to compare the blind spot within each different iterated algorithmic bias mode, alone. We do so by analyzing the class-1-blind spot size. We run experiments with different human action probabilities and record the class-1-blind spot size comparing the blind spot sizes from the first iteration and last iteration. As shown in table 4.14, with higher human action probability, the class-1-blind spot size is higher through all three iterated algorithmic bias modes.

RQ 3.3: Does class imbalanced initialization affect the relevance prediction inequality or Gini coefficient during iterative learning given a fixed iterated algorithmic bias mode?

We wish to compare the inequality induced by each algorithmic bias form, alone. We do so by analyzing the Gini coefficient. We perform the Mann-Whitney U statistical test to see whether the human action will lead to different trends in the inequality of prediction. We run experiments with different human action probabilities, namely 1.0, 0.5 and 0.01, and compare the inequality between the first iteration and last iteration. Table 4.15 shows that higher human action probability leads to high inequality with the filter bias mode.

TABLE 4.15: Results of the Mann-Whitney U test and t-test for the inequality of the computed three forms of iterated algorithmic bias based on the different probability levels of human action. Here, effect size is computed as $ES = (Gini|_{t=0} - Gini|_{t=200})/standard.dev$. Bold means the significance at $p < 0.05$. The more the human reacts to the system, the bigger is the inequality in the prediction.

	Measurement	Filter Bias	Active Learning	Random
$p_{action} = 1$	Mann-Whitney test p-value	**2.5e-14**	**3.8e-9**	0.46
	t-test p-value	**5.2e-35**	**2e-9**	0.68
	effect size	**-1.7**	**1.2**	-0.05
$p_{action} = 0.5$	Mann-Whitney test p-value	**3.5e-26**	**1.84e-21**	0.46
	t-test p-value	**2.5e-14**	**1.8e-14**	0.54
	effect size	**-1.3**	**1.78**	-0.07
$p_{action} = 0.01$	Mann-Whitney test p-value	0.16	0.14	0.45
	t-test p-value	0.17	0.001	0.22
	effect size	0.023	0.18	0.03

On the other hand, active learning significantly decreases the inequality by querying points which are near the learned boundary.

4.3.4 RQ 4: Does human preference towards labeling relevant data affect the boundary?

In research question 3 (RQ 3), we assumed that humans have a prior probability to act which is not dependent on the true label of the item presented. However, in a more realistic world, humans interact with the recommended items, according to their inner preference. In this section, we try to simulate this by assuming that humans have a higher probability to label or rate when the item presented is from relevant class y=1 (see Eq. 3.19). We therefore setup the class-dependent human action probability ratio to 10:1, i.e., $p(action|y = 1) = 10p(action|y = 0)$.

RQ 4.1: Does human preference towards labeling relevant data affect the boundary shift during iterative learning given a fixed iterated algorithmic bias mode?

To answer this question, we run experiments and record the number of points which are predicted to be in class $y = 1$ before and after iterative learning. We first aim to

TABLE 4.16: Results of the Mann-Whitney U test and the t-test comparing boundary shift for the three forms of iterated algorithmic bias with class-dependent human action probability ratio 10:1. The effect size is calculated as $(Boundary|_{t=0} - Boundary|_{t=200})/standard.dev$. The negative effect size for filter bias shows that it decreases the number of points which are predicted to be in class y=1. Random selection increases the number of points, while active learning does not have a significant effect.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**1.8e-13**	0.05	**4.3e-7**
t-test p-value	**6.4e-24**	0.18	**6.1e-13**
effect size	**1.6**	-0.2	**-1.1**

understand how human preference changes the boundary. We do so by performing the Mann-Whitney test and t-test. Table 4.16 shows that the filter bias significantly decreases the number of points predicted to be in class y=1. On the other hand, random selection significantly increases the number of points predicted to be in class y=1. Active learning has no such significant effect. We also compare the effect across different iterated algorithmic bias modes by performing a Kruskal-wallis test. The p-value is 1.87e-21 (< 0.01), which indicates that the bias modes have different effects on the boundary shift (see Figure 4.9).

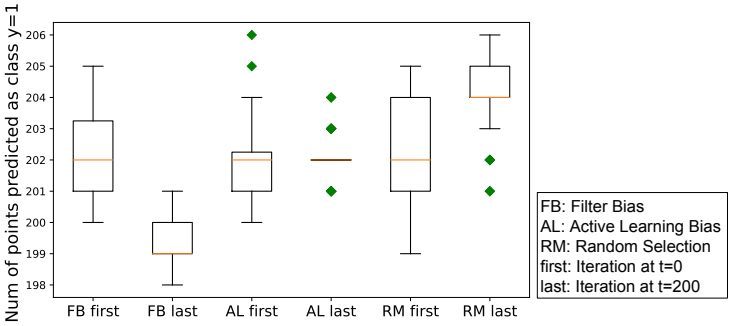


Figure 4.9: The box-plots showing the distribution of the number of points that are predicted to be in class y=1 in the first and last iterations of the iterated learning for three iterated algorithmic biases with human action probability ratio 10:1. Filter bias significantly decreases the number of points predicted to be in class y=1. On the other hand, random selection significantly increases the number of points predicted to be in class y=1. Active learning has no such significant effect.

It is also interesting to compare the effect on boundary shift for different bias modes when the class-dependent human action probability ratios are set as 10:1 and 1:1. Table 4.17 shows the results. Both Filter bias and active learning biases show no significant

TABLE 4.17: Results of the Mann-Whitney U test and the t-test comparing boundary shift for the three forms of iterated algorithmic bias with class-dependent human action probability ratio 1:1 and 10:1. The effect size is calculated as $(Boundary|_{ratio=1:1} - Boundary|_{ratio=10:1})/standard.dev$ at time t=200. Both Filter bias and active learning bias show no significant difference between the two ratios. On the other hand, random selection leads to more points predicted to be in class y=1.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	0.038	0.30	**7.8e-9**
t-test p-value	0.07	0.18	**1.4e-9**
effect size	-0.4	0.04	**-1.28**

difference in boundary shift between the two ratios. On the other hand, random selection has increased the number of points predicted to be in class y=1 when the ratio was 10:1. The results are expected, since filter bias already prefers points from class y=1. Active learning bias shows a smaller effect since this bias already prefers points that are close to the boundary. Random selection with class-dependent human action probability prefer points from class y=1, however randomly. Therefore, it slightly shifts the boundary to class y=0.

RQ 4.2: Does human preference towards labeling relevant data affect the size of the blind spot during iterative learning given a fixed iterated algorithmic bias mode?

To answer this question, we run experiments and record the size of the class-1 blind spot during iterative learning. Recall that the class-1 blind spot indicates how the interaction affects the human's ability to discover items. We first aim to understand how human preference changes the blind spot size before and after iterative learning. We do so by performing the Mann-Whitney test and t-test. Table 4.18 shows that filter bias significantly increases the size of the class-1 blind spot. On the other hand, random selection significantly decreases the size of the class-1 blind spot. Active learning has no such significant effect. We also compare the effect across different iterated algorithmic bias modes by performing a Kruskal-wallis test. The p-value is 3.4e-15 (< 0.01), which indicates that they have different effects on the boundary shift (see Figure 4.10).

TABLE 4.18: Results of the Mann-Whitney U test and the t-test comparing the size of class-1 blind spot for the three forms of iterated algorithmic bias with class-dependent human action probability ratio 10:1. The effect size is calculated as $(Boundary|_{t=0} - Boundary|_{t=200})/standard.dev$. The negative effect size for filter bias shows that it increases the size of the class-1-blind spot. Random selection decreases the blind spot size, while active learning does not have a significant effect.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**1.1e-10**	0.16	**0.0008**
t-test p-value	**1.2e-9**	0.32	**0.002**
effect size	**-1.2**	-0.22	**0.7**

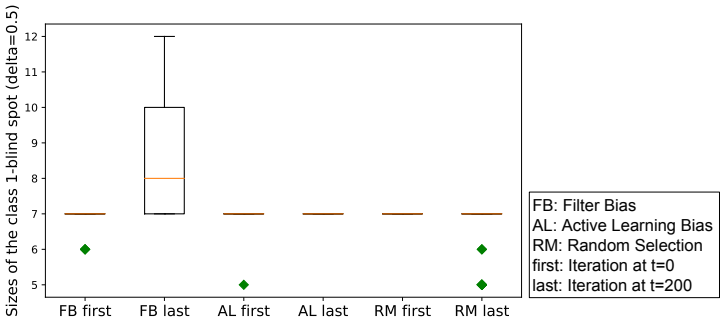


Figure 4.10: The box-plots showing the size of the class-1 blind spot in the first and last iterations of the iterated learning for three iterated algorithmic biases with human action probability ratio 10:1. Filter bias significantly decreases the number of points predicted to be in class y=1. On the other hand, random selection significantly increases the number of points predicted to be in class y=1. Active learning has no such significant effect.

It is also interesting to compare the effect on the class-1 blind spot size with the class-dependent human action probability ratio set to 10:1 and 1:1. Table 4.19 shows the results for the class-1 blind spot size. Both Filter bias and active learning bias show no significant difference between the two ratios. On the other hand, random selection has increased the number of points predicted to be in class y=1. Filter bias already prefers points from class y=1 so they do not show any significant difference. Active learning bias has less effect since it prefers points that are close to the boundary. Random selection with class-dependent human action probability prefers points from class y=1, however it is randomly. Therefore, it slightly shifts the boundary to class y=0.

TABLE 4.19: Results of the Mann-Whitney U test and the t-test comparing boundary shift for the three forms of iterated algorithmic bias with class-dependent human action probability ratio 1:1 and 10:1. The effect size is calculated as $(Boundary|_{ratio=1:1} - Boundary|_{ratio=10:1})/standard.dev$ at time t=200. Both Filter bias and active learning bias show no significant difference between the two ratios. On the other hand, random selection decreases the class-1 blind spot size.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	0.7	1.0	**0.003**
t-test p-value	0.47	1.0	**0.004**
effect size	0.08	0.0	**0.64**

TABLE 4.20: Results of the Mann-Whitney U test and the t-test comparing the inequality of prediction for the three forms of iterated algorithmic bias with class-dependent human action probability ratio 10:1. The effect size is calculated as $(Gini|_{t=0} - Gini|_{t=200})/standard.dev$. The negative effect size for filter bias shows that it increases the inequality. Both active learning and random selection decrease the inequality.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**8.2e-14**	**4.6e-14**	**1.3e-7**
t-test p-value	**3.3e-34**	**1.7e-13**	**2.9e-19**
effect size	**-1.6**	**1.31**	**1.84**

RQ 4.3: Does human preference towards to relevant class affect the inequality or Gini coefficient during iterative learning given a fixed iterated algorithmic bias mode?

To answer this question, we run experiments and record the Gini coefficient before and after iterative learning. We first aim to understand how human preference affects the prediction inequality. We do so by performing the Mann-Whitney test and t-test. Table 4.20 shows that the filter bias significantly increases the inequality. On the other hand, random selection significantly decrease the inequality as well as active learning. We also compare the effect across different iterated algorithmic bias modes by performing a Kruskal-wallis test. The p-value is 3.3e-18 (<0.01), which indicates that they have different effect on the boundary shift (see Figure 4.11).

It is also interesting to compare the effect on the inequality of prediction with the class-dependent human action probability ratio set to 10:1 and 1:1. Table 4.21 shows the results for the inequality of prediction. Both Filter bias and active learning bias show

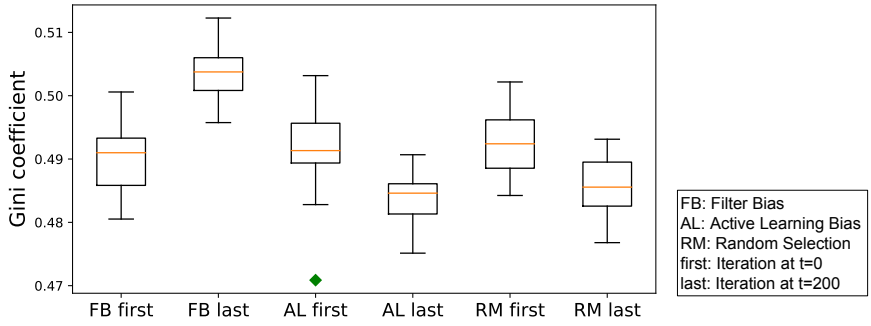


Figure 4.11: The box-plots showing the inequality score at the first and last iterations of the iterated learning for three iterated algorithmic biases with human action probability ratio 10:1. Filter bias significantly increases the Gini coefficient. On the other hand, random selection significantly decrease the Gini coefficient, as well as the active learning.

TABLE 4.21: Results of the Mann-Whitney U test and the t-test comparing the inequality for the three forms of iterated algorithmic bias with class-dependent human action probability ratio 1:1 and 10:1. The effect size is calculated as $(Boundary|_{ratio=1:1} - Boundary|_{ratio=10:1})/standard.dev$ at time t=200. Both Filter bias and active learning bias show no significant difference between the two ratios. On the other hand, random selection has decreased the class-1 blind spot size.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	0.47	0.11	**1.8e-6**
t-test p-value	0.83	0.83	**1.6e-8**
effect size	0.04	0.05	**1.06**

no significant difference between the two ratios. On the other hand, random selection has increased the number of points predicted to be in class y=1. There is no significant difference for the filter bias mode, since it already prefers points from class y=1. Active learning bias has less effect since it prefers points that are close to the boundary. Random selection with class-dependent human action probability prefers points from class y=1, however this occurs randomly. Therefore, it slightly decreases the inequality.

4.4 Experiments on High Dimensional Synthetic Data

We perform similar experiments on 3D, 5D, and 7D synthetic data using a similar data generation method. Our experiments produced similar results to the 2D data. We found that as long as the features are independent from each other, similar results are

TABLE 4.22: Experimental results with the 3D, 5D, 7D and 10D synthetic data set. The effect size is calculated by $(Measurement|_{t=0} - Measurement|_{t=200})/std(\cdot)$. The measurements are the three metrics presented in Section 3.6. We report the paired t-test results. For filter bias mode (FB), the results are identical to those of the 2D synthetic data across all three research questions. Active learning bias (AL) generates the same result as for the 2D synthetic data. Random selection (RM) has no obvious effect, similarly to the 2D synthetic data experiments.

	Bias type	Boundary Shift (p-value, ES)	Blind spot (p-value, ES)	Inequality (p-value, ES)
Statistical test 3D	FB	**(4.3e-22, 2.9)**	**(1.4e-16, -1.2)**	**(1.6e-30, -1.5)**
	AL	**(0.03, 0.34)**	(0.32, -0.19)	**(1.7e-30, 1.9)**
	RM	(1.0, 0.0)	(0.74, 0.04)	(0.87, 0.01)
Statistical test 5D	FB	**(1.4e-17, 2.9)**	**(2e-4, -0.8)**	**(2.4e-30, -1.2)**
	AL	**(0.03, -0.34)**	(0.32, 0.22)	**(9.4e-19, 1.67)**
	RM	(0.63, -0.05)	(0.3, 0.2)	(0.66, -0.04)
Statistical test 7D	FB	**(5.5e-21, 2.2)**	**(1e-7, -0.93)**	**(2.4e-28, -1.2)**
	AL	**(0.01, 0.46)**	(0.05, -0.47)	**(1.5e-15, 1.56)**
	RM	(0.91, -0.01)	(0.09, -0.2)	(0.8, -0.03)
Statistical test 10D	FB	**(8e-15, 1.4)**	**(3e-13, -1.4)**	**(1.8e-13, -1.6)**
	AL	(0.68, -0.09)	(0.5, 0.15)	**(1.8e-15, 1.63)**
	RM	(0.17, 0.17)	(0.1, -0.3)	(0.8, -0.01)

obtained to the 2D case above. One possible reason is that when features are independent, we can reduce them in a similar way to the 2D synthetic data set, i.e., one set of features that are highly related to the labels and another set of features that are non-related to the labels. Another possible reason is that independent features naturally fit the assumption of the Naive Bayes classifier. Finally, we generated a synthetic data with 10 dimensions, centered at (-2,0,0,0,0,0,0,0,0,0) and (2,0,0,0,0,0,0,0,0,0) with zero covariance between any two dimensions. We follow the same experimental procedure as the 2D synthetic data. Table 4.22 shows that the 3D, 5D, 7D and 10D synthetic data leads to similar results to the 2D synthetic data set. In order to avoid repetition, we here only report the results that show how different iterated algorithm bias modes affect the learned model during iterative learning. To conclude, repeated experiments on additional data with dimensionality ranging from 2 to 10 led to the same conclusions as the 2D data set.

4.5 Experiments on the Real-life Data

We follow the same experimental procedure as for the synthetic data set from Section 4.3, but using the MovieLens data (as described in Section 4.2). 200 items are randomly selected as testing set from both classes (like/dislike) for user 547. The testing set is also considered as validation set for the blind spot study. After that, 200 more items from the negative class (dislike) and 300 more items from the positive class (like) are randomly selected to initialize the first boundary. The reason we chose a different proportion from each is to be consistent with the proportions in the whole data set. The iterated learning commences after the initialization and continues for 200 iterations. With the real data, we aim to investigate how different iterated algorithmic biases affect the learned model. Thus, the human action probability is set to $p_{action} = 1$ and the preference to a certain class is not considered.

Visualizing the boundary in high-dimensional data is difficult. However most classifiers produce connected areas for each group [151]. We therefore can employ the number of data points which changed their label after applying the new model to intuitively capture the level of the boundary shift and to understand how the model is affected during the iterated learning process. By also studying the blind spot evolution, we can get a sense of the items that have a very low probability to be shown to the user, because they are part of the blind spot when considering the probability of belonging to class $y = 1$.

4.5.1 Boundary Shift Study

Following the same procedure as in Section 4.3.2, we run experiments for three different forms of iterated algorithmic bias and record the number of points which are predicted to be in class 1 in the first iteration and last iteration. We perform the Mann-Whitney U statistical test to see whether the different iterated algorithmic biases lead to different trends of the boundary shift, we also report the t-test results and effect size. Table

TABLE 4.23: Results of the Mann-Whitney U test and the t-test comparing the boundary shift for the three forms of iterated algorithmic bias with the Movielens data set. The positive effect size indicates that filter bias leads to fewer points predicted to be in class y=1. Random selection and active learning do not have a significant impact.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**7.1e-15**	**0.002**	0.42
t-test p-value	**7.6e-25**	**0.0004**	0.84
effect size	**1.88**	**-0.57**	-0.04

4.23 shows that there is a significant decrease in the number of points predicted to be in class y=1 in the testing set for iterated filter bias. On the other hand, there is no significant difference with active learning bias and random selection. The effect size here is calculated using $(Boundary|_{t=0} - Boundary|_{t=200})/standard.dev$, we will use this setup for the rest of this section. It is also interesting that three different iterated algorithmic bias modes have different results of prediction given similar initialization (see Figure 4.12). All these results are consistent with the results from the synthetic data set.

We conclude that both iterated filter bias and iterated active learning bias have a significantly effect on the boundary shift, while random selection does not have a significant effect. This means that the nature of the model, and hence which items will be judged to be relevant to the user, changes depending on the iterated algorithmic bias, with filtering bias exerting the biggest influence. This kind of phenomenon was found to hold based on all the statistical tests performed in this paper for synthetic and real data.

4.5.2 Blind Spot Size Study

In order to test how different iterated algorithmic bias modes affect the blind spot size, we ran experiments for three different forms of iterated algorithmic bias and recorded the size of the blind spot in the first iteration and last iteration. We performed the Mann-Whitney U statistical test and we also report the t-test results and effect size. Table 4.24 shows that there is a significant increase in the class-1 blind spot sizes in the testing set for iterated filter bias and active learning bias. On the other hand, there is no significant difference

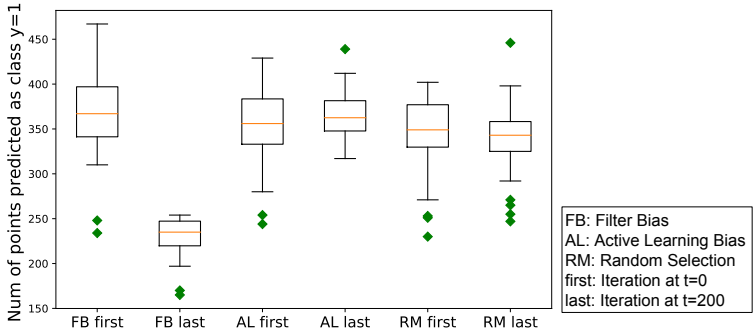


Figure 4.12: The box-plots showing the distribution of the number of points that are predicted to be in class y=1 in the first and last iterations of the iterated learning for three iterated algorithmic biases with the MovieLens data set. Filter bias significantly decreases the number of points predicted to be in class y=1. On the other hand, random selection and Active learning have no such significant effect.

TABLE 4.24: Results of the Mann-Whitney U test and the t-test comparing the size of the class-1-blind spot for the three forms of iterated algorithmic bias with the Movielens data set. The negative effect size indicates that filter bias leads to a bigger blind spot. Both random selection and active learning do not have a significant impact.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**7.0e-15**	**0.0002**	0.15
t-test p-value	**4.2e-26**	**0.0001**	0.46
effect size	**-1.89**	**0.66**	0.16

with random selection (see Figure 4.13). As shown in Figure 4.13, around $150/200 = 75\%$ of items in the relevant testing set are hidden or in the blind spot after interaction.

It is important to note that this impact is significant enough to result in hiding even relevant items from the user. A similar impact was found when the initial labeled (training) set is class imbalanced with more relevant items than non relevant items. **Optional human willingness to label items seems to have a significant impact on the resulting model and thus on which items are judged to be relevant and in turn can be discovered by the user**.

4.5.3 Inequality Study

In order to test how different iterated algorithmic bias modes affect the inequality of predic-

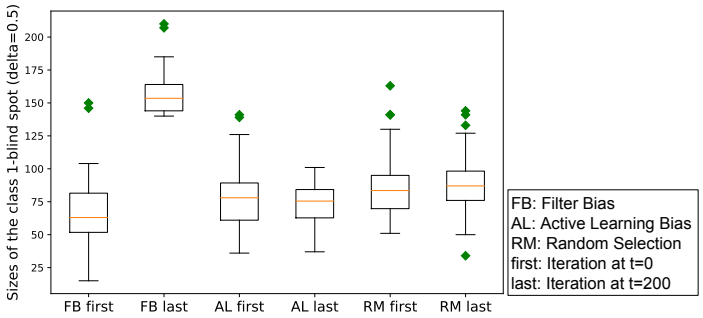


Figure 4.13: The box-plots showing the distribution of the size of the class-1 blind spot in the first and last iterations of the iterated learning for three iterated algorithmic biases with the MovieLens data set. Filter bias significantly increases the size of the class-1 blind spot. On the other hand, random selection and Active learning have no such significant effect.

TABLE 4.25: Results of the Mann-Whitney U test and the t-test comparing the inequality of prediction for the three forms of iterated algorithmic bias with the Movielens data set. The negative effect size indicates that filter bias leads to high inequality of relevance prediction. Both random selection and active learning significant decrease on the inequality.

	Filter Bias	Active Learning	Random Selection
Mann-Whitney test p-value	**7.7e-15**	**1.9e-5**	**0.008**
t-test p-value	**7.4e-25**	**1.7e-9**	**4.4e-5**
effect size	**-1.83**	**0.95**	**0.5**

tions, we ran experiments for three different forms of iterated algorithmic bias and recorded the Gini coefficient in the first iteration and last iteration. We performed the Mann-Whitney U statistical test and we also report the t-test results and effect size. Table 4.25 shows that filter bias leads to a significant increase in the Gini coefficient, while both active learning and random selection result in a significant decrease in the Gini coefficient. It is important to note that both random selection and active learning are used to build an accurate model in machine learning. Therefore, both bias modes decrease the inequality (see Figure 4.14).

We also ran experiments with other 7 users. They all produced similar results (see table 4.26).

4.6 Summary and Conclusions

We investigated three forms of iterated algorithmic bias (filter, active learning, and random baseline) and how they affect the performance of machine learning algorithms by

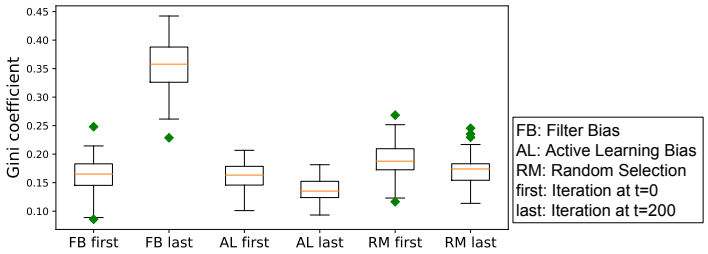


Figure 4.14: The box-plots showing the distribution of the Gini coefficient of the prediction in the first and last iterations of the iterated learning for three iterated algorithmic biases on the MovieLens data set. Filter bias significantly increases the in equality of prediction. On the other hand, random selection and Active learning lead to a significantly decrease in the inequality of prediction.

TABLE 4.26: Top 8 most active users in MovieLens data set and statistical analysis results with paired t-test. The effect size is calculated as $(Measurement|_{t=0} - Measurement|_{t=200}$ $/standard.dev$. Bold means significance, and ES means the effect size. Filter bias has a consistently significant and sizable effect on the three measurements across all 8 users. Random selection has less impact on the boundary shift and blind spot. However it significantly decreases the inequality. Active learning aims to help learn correct boundary, therefore it highly depends on the initial data points. Active learning affects points close to the boundary, thus it has limited effects overall.

	Total movies rated	Positive rated	Negative rated	Bias type	Boundary Shift (p-value, ES)	Blind spot (p-value, ES)	Inequality (p-value, ES)
User ID=547	2391	1409	982	FB	**(7.6e-25, 1.88)**	**(4.2e-26, -1.89)**	**(7.4e-25, -1.83)**
				AL	**(0.0004, -0.57)**	**(0.001, 0.66)**	**(1.7e-9, 0.95)**
				RM	(0.84, -0.04)	(0.46, 0.16)	**(4.4e-5, 0.5)**
User ID=564	1868	1115	753	FB	**(1.0e-14, 1.37)**	**(1.1e-12, -1.45)**	**(1.5e-21, -1.67)**
				AL	**(1.7e-9, -0.99)**	**(3e-9, 0.95)**	**(9e-12, 0.92)**
				RM	**(0.004, -0.48)**	**(0.008, 0.45)**	**(2.6e-6, 0.73)**
User ID=624	1735	1043	692	FB	**(2.3e-7, 0.8)**	**(2.7e-7, -1.28)**	**(8.2e-26, -1.7)**
				AL	**(8.4e-11, -1.28)**	**(7.8e-11, 1.29)**	**(7.5e-17, 1)**
				RM	**(2.8e-5, 0.64)**	**(1.6-5, 0.7)**	**(8e-6, 0.64)**
User ID=15	1700	857	843	FB	**(3.4e-18, 1.4)**	**(2.3e-18, -1.5)**	**(2.6e-30, -1.76)**
				AL	**(4.3e-8, 0.93)**	**(7.4e-9, 0.9)**	**(0.0003, 0.35)**
				RM	**(1.6e-6, 0.73)**	**(2.1e-6, -0.7)**	(0.16, -0.17)
User ID=73	1610	1016	594	FB	**(1.8e-8, 1.02)**	**(1.6e-8, -1.0)**	**(7.6e-36, -1.81)**
				AL	**(1.7e-12, -1.5)**	**(5.4e-11, 1.46)**	**(9.3e-24, 1.42)**
				RM	**(2e-7, -0.9)**	**(5.6e-7, 0.85)**	**(6.4e-7, 0.66)**
User ID=452	1340	613	727	FB	**(0.001, 0.46)**	**(0.0006, -0.7)**	**(1.1e-12, -1.31)**
				AL	**(2.8e-8, 1.2)**	**(5.9e-10, -1.24)**	(0.78, 0.34)
				RM	**(0.0101, 0.5)**	**(0.009, -0.52)**	(0.7, 0.04)
User ID=468	1291	795	496	FB	**(5.2e-10, 1.18)**	**(8.8e-12, -1.29)**	**(3.3e-26, -1.74)**
				AL	**(1.8e-18, -1.6)**	**(3.1e-18, 1.61)**	**(1.9e-26, 1.67)**
				RM	**(5.6e-12, -1.36)**	**(3.2e-13, 1.44)**	**(1.1e-13, 1.15)**
User ID=380	1063	620	443	FB	**(3.2e-11, 1.22)**	**(2.6e-11, -1.26)**	**(2.3e-27, -1.76)**
				AL	**(1.36e-13, -1.32)**	**(1.13e-12, 1.28)**	**(4.7e-15, 0.96)**
				RM	**(1.3e-9, -0.96)**	**(1.53e-5, 0.89)**	**(0.002, 0.7)**

formulating research questions about the impact of each type of bias. Based on statistical analysis of the results of several controlled experiments using synthetic and real data, we found that (see the overall synthesis of findings in Table 4.27 and 4.28):

1. The three different forms of iterated algorithmic bias (filter, active learning, and random selection, used as query mechanisms to show data and request new feed-

back/labels from the user), **do affect algorithm performance** when fixing the human interaction probability to 1.

2. Different initial class imbalance in the training data used to generate the initial relevance boundary, significantly affect the machine learning algorithm's results for all three forms of iterated algorithmic bias, **impacting boundary shift, heterogeneity of predicted relevance, and hidden relevant items (class 1-blind spot)**.

3. Iterated filter bias has a more significant effect on the class-1-blind spot size compared to the other two forms of algorithmic biases. **This means that iterated filter bias, which is prominent in personalized user interfaces, can limit humans' ability to discover data that is relevant to them**.

4. The iterated learning framework is effective for analyzing the impact of iterated algorithmic bias in human-algorithm interaction.

Our findings indicate that the relevance blind spot (items from the testing set whose predicted relevance probability is less than 0.5) amounted to 4% of all relevant items when using a content-based filter that predicts relevant items. A similar simulation using a real-life data set found that the same filter resulted in a blind spot size of 75% of the relevant testing set. Future work will consider the following three directions: 1) Taking into account additional parameters and configurations when testing the impact of iterated algorithmic bias and human interaction on ML models, including: (A) Changing the number of items recommended in top N relevant item recommendation lists and (B) Applying different types of AL (we only investigated uncertainty-based AL); 2) Human experiments that study research questions that are similar to the ones formulated for the simulated experiments; 3) In our study, we considered three iterated algorithmic biases modes, there are more possible iterated algorithmic biases modes, such as active-filter bias (Combining active learning and filter bias, i.e., querying data which are close to the center of each class); 4) Here, we used two different models to model human action, human reaction is more complicated. Future work would be to develop more realistic human reaction models.

TABLE 4.27: Summary of Research Question (RQs) and findings for synthetic data

RQ	Sub Questions	Main Findings
1	• RQ1.1: Do different forms of iterated algorithmic bias have different effects on the boundary shift? • RQ1.2: Do different iterated algorithmic bias modes lead to different trends in the inequality of predicted relevance throughout the iterative learning? • RQ1.3: Does the iterated algorithmic bias affect the size of the class-1-blind spot?	• Filter bias reduces the # of items predicted as class y=1 and increases the blind spot size, which indicates it will limit the users' ability to discover new items. • Filter bias increases the inequality of prediction, which leads to even more inequality. • Random selection and active learning bias show no/little effect on the blind spot.
2	• RQ2.1: Does class imbalanced initialization affect the boundary learned during iterative learning? • RQ2.2: Does class imbalanced initialization affect the blind spot size during iterative learning? • RQ2.3: Does class imbalanced initialization affect the inequality of predictions during iterative learning?	• Highly imbalanced class initialization leads to a bigger difference between the learned boundary and the ground-truth boundary. • When the initialization ratio is 10:1 (class 1: class 0), all three bias forms increase the class-1 blind spot size, indicating that imbalanced initialization leads to even more limitation for users to discover new items. • When the initialization ratio is 1:10 (class 1: class 0), filter bias results in increasing the blind spot size. Random selection and active learning bias decrease the blind spot size.
3	• RQ3.1: Does human action affect the boundary shift during iterative learning? • RQ3.2: Does human action affect the class-1-blind spot size during iterative learning given a fixed iterated algorithmic bias mode? • RQ3.3: Does class imbalanced initialization affect the relevance prediction inequality (or Gini coefficient)?	• Human action affects the boundary shift more with iterated filter bias than with the other two forms of bias. • For both random selection and active learning bias, the number of points predicted as relevant converges to the ground truth boundary when there is a high probability of human action. • Filter bias tends to diverge from the ground truth boundary with high human action probability. • The more humans react to the recommender system, the higher the impact of each iterated algorithmic bias mode.
4	• RQ4.1: Does human preference towards labeling relevant data affect the boundary shift during iterative learning? • RQ4.2: Does human preference towards labeling relevant data affect the size of the blind spot during iterative learning? • RQ4.3: Does human preference towards to relevant class affect the inequality or Gini coefficient during iterative learning?	• For both filter bias and active learning, the number of points predicted as class y=1, with and without the class-dependent human action probability, show no significant difference. • For both filter bias and active learning, the size of the class-1 blind spot, with and without the class-dependent human action probability, have no significant difference. • Random selection decreases the class-1 blind spot size, and increases the number of points predicted as class y=1, with the class-dependent human action probability.

TABLE 4.28: Summary of Research Question (RQs) and findings for real data

Research Question	Main Findings
• How do different iterated algorithmic bias modes affect the boundary shift?	• Both iterated filter bias and iterated active learning bias have a significantly effect on the boundary shift, while random selection does not have a significant effect, indicating that the nature of the model, and hence which items will be judged to be relevant to the user, changes depending on the iterated algorithmic bias, with filtering bias exerting the biggest influence.
• How do different iterated algorithmic bias modes affect the blind spot size?	• There is a significant increase in the class-1 blind spot size in the testing set for iterated filter bias and active learning bias. On the other hand, there is no significant difference with random selection.
• How do different iterated algorithmic bias modes affect inequality of prediction?	• Filter bias leads to a significant increase in the Gini coefficient, while both active learning and random selection show a significant decrease in the Gini coefficient.

CHAPTER 5

DEBIASING COLLABORATIVE FILTERING RECOMMENDER SYSTEMS

Recommender Systems (RSs) are widely used to help online users discover products, books, news, music, movies, courses, restaurants, etc. Because a traditional recommendation strategy always shows the most relevant items (thus with highest predicted rating), traditional RS's are expected to make popular items become even more popular and non-popular items become even less popular which in turn further divides the haves (popular) from the have-nots (unpopular). Therefore, a major problem with RSs is that they may introduce biases affecting the exposure of items, thus creating a popularity divide of items during the feedback loop that occurs with users, and this may lead the RS to make increasingly biased recommendations over time. In this chapter, we view the RS environment as a chain of events that are the result of interactions between users and the RS. Based on that, we propose several debiasing algorithms during this chain of events, and evaluate how these algorithms impact the predictive behavior of the RS, as well as trends in the popularity distribution of items over time. We also propose a novel blind spot awareness matrix factorization algorithm to debias the RS. Results show that the proposed propensity matrix factorization achieved a certain level of debiasing of the RS while active learning combined with the propensity MF achieved a higher debiasing effect on recommendations. Our proposed blind spot aware matrix factorization also achieved a certain level of debiasing of the RS.

5.1 Introduction

The goal of a RS is to infer user preferences, given the user's previous ratings, and to predict which items the user might like. Modern RSs generally aim to discover a stable

relationship between users and items. This may lead to a situation in which users only see a narrow subset of the entire range of available recommendations, a phenomenon known as the 'filter bubble' [75]. The relationship between users and items is, however, time dependent. This is because if the RS predicts that some items may not be of interest to the user, these items may end up never actually being seen by the user. This presents a significant problem for RSs: we might know why a user likes an item, but we do not know why an item is not liked by the user. Is it not liked by a user because the user does not like it, or is it simply because the user has not seen the item in the RS results? Furthermore, if we assume the RS will continue to recommend items to users based on biased ratings, and that users will respond to these recommendations, the RS will slowly learn to return increasingly similar items in its results. In other words, the RS will begin to systematically limit the users' ability to discover more items [152]. In this chapter, we propose to model how iterated biases evolve from the continuous user-RS feedback loop, and propose to develop a series of different debiasing strategies.

Unlike the existing work on debiasing, reviewed in Sec 2.6, we aim to propose strategies to reduce the iterated bias that occurs during the interactions between users and the RS without the introduction of strong assumptions. Moreover, in contrast to the aforementioned research, we propose algorithms that consider the RS to be a chain of events, and then focus on debiasing the iterated bias introduced by these interactions by using an estimated propensity score, with and without an active learning strategy. We also employ the Gini coefficient and the blind spot score to quantify how the interaction affects the users' ability to discover new items. Table 5.1 shows the related work and a comparison with our work. Part of this work is also reported in [153], but only evaluate the methods on synthetic data.

5.2 Objectives and Contributions

Viewing a RS as a continuous chain of events in which users actively interact with the output of a RS, we propose three debiasing algorithms for RSs:

TABLE 5.1: Related work on Debiasing.

Related work	Main Goal	Bias Models Proposed	Debiasing Strategies	Evaluation Metrics
Hu et al. 2008, [120]	Understanding implicit feedback in RS	N/A	N/A	MAE and NDCG
Schnabel et al. 2016, [121]	Debiasing learning and evaluation in RS	Exposure model	Inverse propensity weighting	MSE and NDCG
Liang et al. 2017, [122]	Debiasing implicit feedback in RS	Exposure bias	Incorporating user exposure model into RS	Recall, NDCG and MAP
Chaney et al. 2017, [126]	Understanding feedback loops in RS	Homogenization of user behavior	N/A	Homogeneity score
Sinha et al. 2016, [129]	Debiasing feedback loops in RS	$R_{obs} = R_{true} + R_{recom}$	Deconvolving feedback impact	ROC
Nasraoui and Shafto 2016, [77]	Understanding iterative algorithmic biases in machine learning	Iterated algorithmic bias	Reactive learning and antidote	Machine learning performance
Sun et al. 2018, [152]	Modeling and understanding bias	Iterated algorithmic bias	N/A	Gini coefficient and blind spot size
Shafto and Nasraoui. 2016, [130]	Modeling and understanding bias and its connection to human behavior	Dynamic interaction between machine learning systems and humans	Cognitive models	Human's learning behavior
This Work	Understanding and debiasing iterative bias in RS	Propensity model	Inverse propensity weighting, active learning and blind spot aware Matrix Factorization	MAE, RMSE, Gini coefficient and blind spot score

1) a unified recommendation and active learning strategy (active recommendation) during the interaction between users and the RS algorithm, with the goal of reducing recommendation uncertainty, while at the same time ensuring the integrity of the algorithm‘s performance;

2) an exposure-based collaborative filtering recommendation model that is also combined with an active recommendation to further debias the RS;

3) a blind spot aware MF, which takes into account and counteracts the blind spot caused during the learning phase of the RS.

5.3 Propensity and Active Learning

We start by summarizing the notation used in this chapter. All sets and parameters are defined as follows and are based on similar notation given in [121, 122]:

$R_{u,i}$: An integer which indicates the rating giving by user u to item i.

$O_{u,i}$: A binary value which indicates that user u provided a rating for item i to the system, $[O_{u,i} = 1] \rightarrow [R_{u,i} \quad is \quad observed]$.

$P_{u,i}$: Propensity: The probability of observing an entry. $P_{u,i} = P(O_{u,i} = 1)$.

N_U: The total number of users.

N_i: The number of users who rated item i.

5.3.1 Propensity

Observational recommendations contain two sources of information: the items which the user can see, and the user's recorded preference toward those items. *Propensity* refers to the probability of observing a rating $R_{u,i}$. In a real-world application, what the user sees is highly subject to selection biases. For example, users rarely rate movies that they dislike since they may not have seen these movies. Another example would be an advertisement recommendation system that always shows ads that are predicted to be relevant to the user. This bias is expected to get stronger as a result of the iterative interaction between users and the recommendation outputs.

Propensity is well studied in causal inference [154]. Why do we consider recommending items to users to be a causal inference problem? Suppose, the items are movies and users are asked to rate movies they have seen. In the prediction stage, the RS is trying to answer the question: "Will the user like this movie if he or she watches it?". From the perspective of causality, this is a similar question about an intervention or treatment: "What would the rating be if the user is forced to watch the movie?" The item here is the role of the treatment; and the rating is the role of the response to the treatment [121]. The

propensity-weighted approaches are different from traditional approaches. A traditional approach builds a model based on observed ratings, and then uses the model to predict the ratings of unobserved items. This kind of approach usually incorporates the assumption that users typically watch and rate movies at random. This assumption does not hold under the conditions of the iterative user-RS interaction.

Recent research on RSs began to take into account the role of item propensity, where user exposure to an item in a RS is viewed as analogous to exposing a patient to a treatment in a medical study [121, 122]. Thus, propensity is also treated as a causal inference; propensity indicates how probable it is that a new treatment can or will be exposed to a patient. In both cases, the studies try to infer the causal effect based on current results (whether it be the effect of a new treatment, or a new item in a RS).

Existing approaches for RSs generally under-weight items that are not rated in the system. It is, however, difficult to determine whether an item is not rated by a user because the user does not like the item, or because the user has not seen the item as a result of the inherent selection bias in the RS [122]. The main idea behind a propensity-based MF is to under-weight the unrated item for recommendation and up-weight the unpopular item by bringing into the objective function of the model an Inverse Propensity Score, as follows [121]:

$$\underset{V,M}{argmin} \sum_{O_{u,i}=1} \frac{||R - V^T M||^2}{P_{u,i}} + \lambda(||V||_F^2 + ||M||_F^2) \tag{5.1}$$

Here $P_{u,i}$ represents the probability that a user u will see item i, and is also referred to as propensity score. V and M are the two latent factors in the MF.

Estimating propensity

A simple way of estimating propensity is to use a popularity score [122]. This assumes that $O_{u,i}$ follows a Bernoulli distribution, i.e., $O_{u,i} \sim Bernoulli(\rho_i)$. Note that the propensity score is fixed across users in this case, i.e., $P_{u,i} = \hat{\rho}_i$. Given a rating matrix, the popularity of an item is the proportion of items exposed to certain users among all the users (essentially, the proportion of users who have rated the item relative to all users).

Another way of estimating propensity is to assume that $O_{u,i}$ follows a Poisson distribution [122], or $O_{u,i} \sim Pois(\pi_u^T \gamma_i)$. π_u and γ_i are the two latent factors of the Gamma prior. Given a ratings matrix, this method assigns a value of 1 to an item that was rated by a user in the observational matrix O, and 0 if it is not rated. By factoring this observational matrix, we get the propensity scores of all user and item pairs (see Eq. 5.2)

$$P_{u,i} = 1 - P(O_{u,i} = 0|\pi_u, \gamma_i) \approx 1 - exp\left\{-E[\pi_u^T \gamma_i]\right\}. \tag{5.2}$$

5.3.2 Active Learning

Active Learning (AL) is a special case of semi-supervised learning in which the system has the ability to interactively prompt users to label (or rate) items in order to improve the accuracy of the model [78]. One of the advantages of AL is that the targeted knowledge the system acquires helps accelerate the speed at which the system learns the model. One way to implement AL is to actively prompt users to rate items that have been underweighted by the RS in order to improve the quality of the ratings of this subset of items. In a rating system with range from R_{min} to R_{max} , the active learning can be formalized directly as follows [155]: Select the next item x_{act} that satisfies

$$x_{act} = \underset{x_i}{argmin}[\theta - \hat{y}|x_i]. \tag{5.3}$$

Here, $\hat{y}$ is the rating predicted by the RS given an item x_i. θ controls the degree of active learning, ranging from the midrange rating of $0.5(R_{min} + R_{max})$ to the maximum rating R_{max}.

It can be seen that $\theta = R_{max}$ recovers pure recommendation (select the most relevant item, hence the item with highest predicted rating), while $\theta = 0.5(R_{min} + R_{max})$ recovers pure active learning (select the item with most uncertain relevance to the user based on the predicted rating, hence an item that is far from both being very relevant ($\hat{y} = R_{max}$) and very non-relevant ($\hat{y} = R_{min}$)). Cognitive experiments have recently shown that an active recommendation system can cover a wider choice of items, while maintaining the accuracy of the results [155].

5.3.3 Proposed Methods to Debias Recommender Systems

We first introduce our interactive recommender system framework, which considers a RS as a continuous chain of events. First the initial RS suggests items to each user based on the initial training ratings and it is assumed that the users have 100% agreement with the recommendation. We then retrieve the true ratings from our masked ratings and add to the new training ratings. After that, a new recommendation based on new training data will be issued. This interaction will continue until a maximal number of iterations is reached. Algorithm 1 shows the details of our interactive recommender system with human in the loop.

Algorithm 1: Interactive Recommendation System with the Human-Recommender System Feedback Loop Debiasing Mechanism

Data: Rating matrix R'_{ui}, λ, Learning rate η, $MAX_{interation}$, Iterations=0, Size of selection
Result: MAE, RMSE, Gini Coefficient
The system trains the initial Matrix Factorization model and computes predictions $\hat{R}$;
while *Iterations ¡ Max Feedback Loop Iterations* **do**
  1 for all users u in the system {
    1.1. The system selects top-N items to recommend from the predicted ratings $\hat{R}$ based on a specialized recommending strategy;
    1.2. User u picks the selected top-N items and gives rating $R^{new}_{u,i}$ (from ground-truth complete data) for each item i;
    }
  2. The system records the popularity $P_i = N_i/N_U$ after the new ratings are taken in.
  3. The system records the metrics such as the RMSE and the Gini index of the popularity given the current rating matrix;
  4. The system retrains the model with the new rating matrix using *steps* gradient descent updates (Eq. 5.4 or Eq. 5.8 depending on the recommendation strategy chosen, and recomputes the predictions).
  5. Iterations++
end

We propose several recommendation strategies based on common latent factor-based algorithms to simulate real-life user-recommendation system interaction.

Conventional MF:

This model is trained using conventional MF (same as Eq. 5.1 with $P_{u,i} = 1$), and

the system always selects the top predicted item for each user, and adds it to the next (new) training set. In other words, there is no active learning.

Conventional MF + Active Learning:

This model is trained using conventional MF (Eq. 5.1 with $P_{u,i} = 1$), but the system selects the active recommendation items with $\theta = 4.5$ for each user in Eq. (5.3). $\theta = 4.5$ is chosen to be between $\theta = 0.5(R_{min}+R_{max}) = 3$ (pure AL) and $\theta = R_{max} = 5$ (pure recommendation).

Popularity Propensity MF:

This model is trained with propensity MF (Eq. 5.1) [121]. The propensity $P_{u,i}$ is estimated based on popularity. The system always selects the top predicted item for each user, and adds it to the next (new) training set. In other words, there is no active learning.

Popularity Propensity MF + Active Learning:

This model is trained with propensity MF (Eq. 5.1) [121]. The propensity $P_{u,i}$ is estimated using popularity. The system selects the active recommendation items with $\theta = 4.5$ for each user in Eq. (5.3).

Poisson Propensity MF:

The model is trained with propensity MF (Eq. 5.1). Here the propensity $P_{u,i}$ is estimated based on Poisson MF (5.2) [122] on the exposure matrix. The system always selects the top predicted item for each user, and adds it to the next (new) training set. In other words, there is no active learning.

Poisson Propensity MF + Active Learning:

The model is trained with propensity MF (Eq. 5.1). Here the propensity $P_{u,i}$ is estimated using Poisson MF (Eq. 5.2) [122] on the exposure matrix and the system selects the active recommendation items using (Eq. 5.3) with $\theta = 4.5$ for each user (again chosen to be between $\theta = 0.5(R_{min} + R_{max}) = 3$ (pure AL) and $\theta = R_{max} = 5$ (pure recommendation)).

In this preliminary work, we use both popularity and Poisson Matrix Factorization (PMF) to estimate the propensity. To minimize the objective function, we use stochastic gradient descent, which has been used successfully to solve MF with big datasets. For a given training rating r_{ij}, the updates for V_u and M_i can be shown to be:

$$\begin{aligned} V_u &\leftarrow V_u + \eta(2e_{ui}M_i - \lambda V_u) \\ M_i &\leftarrow M_i + \eta(2e_{ui}V_u - \lambda M_i). \end{aligned} \tag{5.4}$$

Here $e_{ui} = \frac{(\hat{r}_{ui} - V_u^T M_i)}{P_{u,i}}$, $\hat{r}_{ui}$ is the predicted rating and η is the learning rate for gradient descent. With a proper choice of step size, gradient descent converges to a local minimum. Propensity $P_{u,i}$ is computed as follows:

$$P_{u,i} = \begin{cases} 1.0 & for\ Conventional\ MF \\ N_i/N_U & for\ Popularity\ Propensity\ MF \\ 1 - P(O_{u,i} = 0|\pi_u, \gamma_i) & for\ Poisson\ Propensity\ MF \end{cases} \tag{5.5}$$

This means that $P_{u,i} = 1$ for the first two (Conventional MF, with or without AL) recommendation strategies. $P_{u,i}$ is estimated using popularity for Popularity Propensity MF (with or without AL) strategies and it is estimated using Eq. 5.2 for Poisson Propensity MF (with or without AL) strategies.

In addition to the above proposed debiasing strategies above, a seventh algorithm called **Blind Spot Aware Matrix Factorization** is introduced. In this chapter, we define the blind spot size as the number of item with a predicted ratings $\hat{R_{u,i}}$ that is smaller than a threshold δ , i.e., $\mathbf{D_\delta^u} = \{\mathbf{i} \in I \mid \hat{R}_{u,i} < \delta\}$. Note that because each user has their own blind spot, we define a threshold for each user. The threshold is used to set a percentile cut-off for each user, 95% in our experiments. Therefore, We define the blind spot for user u as

$$\mathbf{D_\epsilon^u} = \{\mathbf{i} \in I \mid \hat{R}_{u,i} < max_{u,i}(\hat{R}_{u,i}) * \epsilon\}. \tag{5.6}$$

Here ϵ is a cut-off which controls the threshold.

Our proposed Blind Spot Aware MF tries to limit the blind spot when trying to optimize the cost function of the conventional matrix factorization. The cost function for

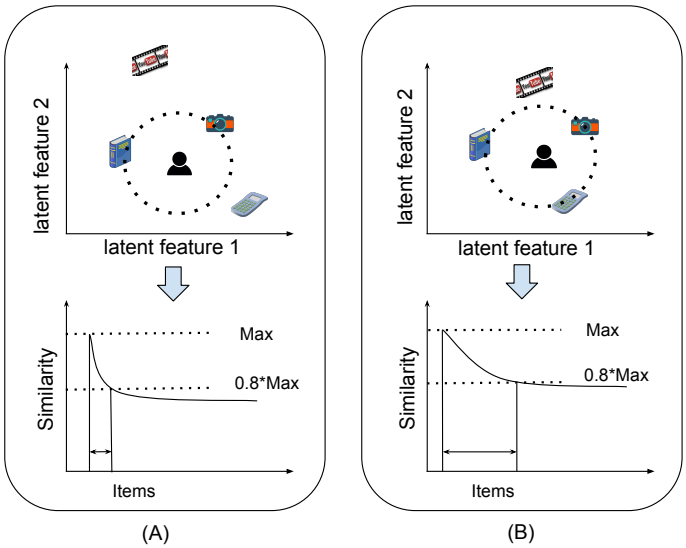


Figure 5.1: Conventional Matrix Factorization vs. Blind Spot Aware Matrix Factorization. Figure (A) indicates Conventional Matrix Factorization, in which the algorithm aims to find items that are close to users through differentiation. The top of Figure (A) indicates how items are distributed around a user in the latent space under Conventional MF, the bottom of (A) shows the similarity between the user and all items in latent space. Figure (B) shows how the proposed Bias-aware Matrix Factorization finds items close to users while keeping the items that are close to each other in the latent space. The top of Figure (B) indicates how items are distributed around a user in the latent space , the bottom of (B) shows the similarity between the user and all items in latent space.

blind spot aware MF is as follows:

$$J = \sum_{u,i \in R} ||r_{u,i} - V_u^T M_i||^2 + \frac{\lambda}{2}(||V_u||^2 + ||M_i||^2) + \underbrace{\frac{\beta}{2}||V_u - M_i||^2}_{Blind\ \ Spot\ \ Aware\ \ Term}. \tag{5.7}$$

To minimize the objective function, we use stochastic gradient descent, which has been used successfully to solve MF for CF with big datasets. For a given training rating r_{ij}, the updates for V_u and M_i can be shown to be:

$$\begin{aligned} V_u &\leftarrow V_u + \eta(2e_{ui}M_i - \lambda V_u - \beta(V_u - M_i)) \\ M_i &\leftarrow M_i + \eta(2e_{ui}V_u - \lambda M_i + \beta(V_u - M_i)). \end{aligned} \tag{5.8}$$

Here $e_{ui} = \hat{r}_{ui} - V_u^T M_i$, $\hat{r}_{ui}$ is the predicted rating, and η is the learning rate for gradient descent. With a proper choice of step size, gradient descent converges to a local minimum.

Intuition Behind Blind Spot Aware MF: Conventional Matrix Factorization aims to predict ratings by matching the existing ratings through some latent factors. The RS tries to differentiate different items for each user as much as possible. The distance between a user and all items are very unevenly distributed compared to Blind Sport Aware Matrix Factorization as shown in Figure 5.1. Therefore, given the same proportional range of similarity, blind spot aware has more items to explore with high relevant score. For example, Conventional MF have very limited choices to recommend to users given the predicted ratings range[0.8*Max, Max]. 'Max' is the maximum of the predicted ratings on items for a user.

On the other hand, Blind Spot Aware Matrix Factorization tries to match the existing rating, bringing items close to each other so that each user has a higher chance to explore more items. For example, the overall blind spot size is decreased (see figure 5.1). As shown in figure 5.1, conventional MF has a very uneven similarity distribution comparing with blind spot aware MF. While, Blind Spot Aware MF has rich candidates of items to choose from. Most importantly, those candidate items are all highly related with the user in the latent space. In these cases, blind spots are partially related with the novelty and diversity in a recommender system (see Section 5.4). Novelty aims to recommend items that are different from previously rated item, while diversity intends to recommend items that are different from each other in the recommended list [156].

5.4 Experimental Evaluation

5.4.1 Data Sets

5.4.1.1 Synthetic Data

We use item response theory to generate a sparse rating matrix $\boldsymbol{R}_{u,i}$ using the model proposed in [157]. Assume a_u to be the center of user u's rating scale, and b_u to be the rating sensitivity of user u. Finally let t_i be the intrinsic score of item i. We generate a

user-item rating matrix as follows:

$$\boldsymbol{R}_{u,i} = L[a_u + b_u t_i + \eta_{u,i}] \tag{5.9}$$

where $L[\omega]$ is the discrete level function, assigning a score in the range 1 to 5: $L[\omega] = \max(\min(\text{round}(\omega), 5), 1)$ and $\eta_{u,i}$ is a noise parameter. In our experiments, we draw $a_u \sim N(3.4, 1)$, $b_u \sim N(0.5, 0.5)$, $t_u \sim N(0.1, 1)$, and $\eta_{u,i} \sim \epsilon N(0, 1)$; where N is a standard normal density, and ϵ is a noise parameter, we set up $\epsilon = 0.5$. We generate a rating matrix R with 500 users and 500 items, therefore we have 250,000 ratings in total. To check the quality of our synthetic ratings, we simply compare the distribution of ratings from our synthetic data with the original MovieLens 100k dataset. Figure 5.2 shows the comparison of those two rating distributions. The Pearson correlation coefficient between the two distributions is 0.98 [158].

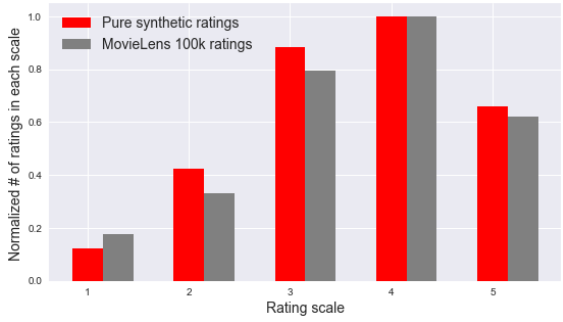


Figure 5.2: Distribution of synthetic ratings and MovieLens 100k rating data. The x-axis is the rating scale, ranging from 1 to 5. The y-axis is the number of ratings in each scale normalized by the maximum.

5.4.1.2 Semi-synthetic Data

We also conduct our experiments on a semi-synthetic dataset. The ML100K dataset provides 100K ratings for 1683 movies by 944 users [159]. To allow a long term user and machine learning system interaction, we complete these partial ratings using standard matrix factorization. We then adjust those ratings to match a more realistic rating distribution [p1, p2, p3, p4, p5] for ratings 1 to 5 as given in [160] as follows: we assign the bottom

p1 fraction of the entries by value in the completed matrix a rating of 1, and the next p2 fraction of entries by value a rating of 2, and so on. Hyper-parameters (rank d and L2 regularization λ for the standard Matrix Factorization) were chosen by using an 80-20 train-test split of the 100K ratings, and maximizing the accuracy of the completed matrix on the test set.

5.4.2 Metrics

In order to assess the accuracy of the prediction of the RS during the interactive recommendation, we compute the Root Mean Square Error (RMSE). To check the impact of the debiasing mechanism, we compute the Gini coefficient of the popularity scores of all items. The Gini coefficient is used to measure the inequality of a distribution [139]. Let P_i be the popularity of each item after training the model. For a population with n values P_i, $i = 1$ to n, that are indexed in non-decreasing order ($P_{(i)} \leq P_{(i+1)}$), the Gini coefficient can be calculated as follows [139]:

$$G = (\frac{\sum_{i=1}^{n}(2i - n - 1)P_{(i)}}{n\sum_{i=1}^{n} P_{(i)}}). \tag{5.10}$$

The higher the Gini coefficient, the more unequal are the values in the dataset. The Gini coefficient of the popularity of items shows how the recommendation system's output affects the exposure distribution of items. Traditional RS's are expected to make popular items become even more popular and non-popular items become even less popular because a traditional recommendation strategy always shows the most relevant items (thus with highest predicted rating), which further divides the haves (popular) from the have-nots (unpopular).

We expect to see the debiasing mechanism lead to different trends in comparison with a traditional recommendation process that shows the top items to users after each iteration of the model learning process. The Gini coefficient of the popularity of items shows how the recommendation system's output affects the exposure distribution of items. Traditional RS's are expected to make popular items become more popular and non-popular items

become even less popular because a traditional recommendation strategy always shows the most highly predicted items.

Another way to measure the debiasing effectiveness is to check the **blind spot size** (see Eq. 5.6). Modern recommendation systems suffer from two important issues: filter bubbles [75], and blind spots [77, 130, 152] (see Fig. 5.3). If the blind spot size is too big, users only have limited ability to discover possibly interesting items. However, if the filter bubble size is too big, users are limited to discover only certain types of items.

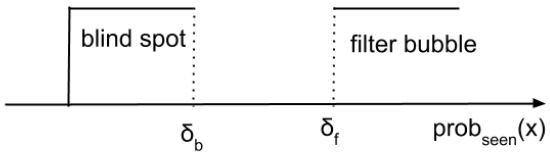


Figure 5.3: The blind spot and filter bubble in a recommender system. Here, δ_b represents the threshold up to which items that have a lower probability of being seen will not be seen by users. On the other hand, items with probability of being seen higher than δ_f will always be seen by users.

In this work, we check how many times an item falls into the blind spot based on the predicted ratings across all users who have not rated or seen this item yet. We use $\mathcal{R}_u = \{i \in I | user \quad u \quad has \quad rated \quad item \quad i\}$ as the observed ratings for user u. Therefore the BL score for an an item i is:

$$BL^{(i)}_{score} = \frac{\sum\limits_{u \in U} |i \in D^u_\epsilon \quad and \quad i \notin \mathcal{R}_u|}{\sum\limits_{u \in U} |i \notin \mathcal{R}_u|} \tag{5.11}$$

The numerator is the total number of users who have not rated item i and have item i in their blind spot based on the prediction. The denominator is the total number of users who have not rated item i. For example, if an item falls into the blind spot 100 times based on Eq. 5.6 across all 400 users who have not rated this item, then the BL score for this item will be $100/400 = 0.25$.

5.4.3 Method

For each data set, we randomly select a certain proportion of ratings to initialize our recommender model. We randomly select 25 ratings for each user for the pure synthetic data. For the semi-synthetic data, we randomly select 50 ratings for each user. After this, we leave out 20% of the entire rating matrix as testing set. Note that the test set is later fixed in both cases, and will not be changed. All other ratings are considered as candidate ratings: they are used to simulate the feedback loop of RS and human interaction, and will be added based on the selection mechanism of the recommendation strategy in each iteration/loop. Figure 5.4 shows the approach for splitting our data. We record the RMSE, MAE, blind spot score and the Gini coefficient of predicted item popularity of each testing set. Figure 5.4 shows the splitting of our data.

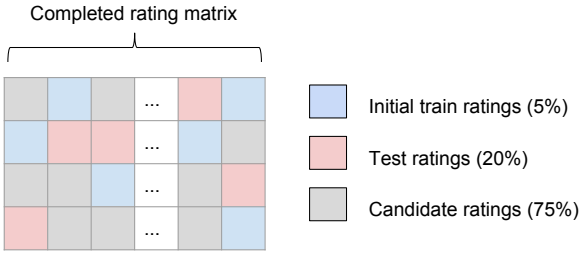


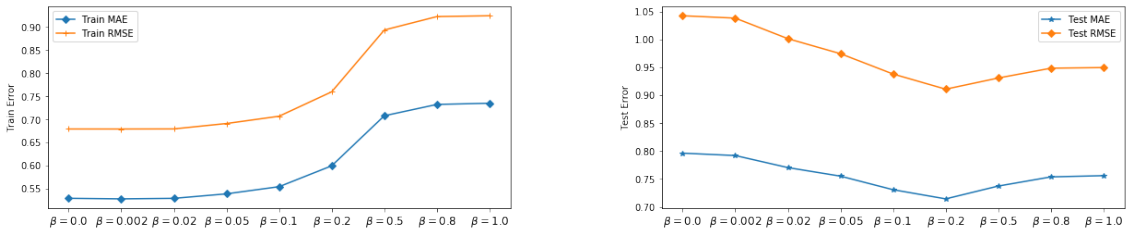
Figure 5.4: The splitting of the completed rating matrix. The completed rating matrix is split into 3 parts: 1) Initial ratings; 2) Test ratings; and 3) Candidate ratings. The candidate ratings will be added to the training ratings when queried by the system.

For both data sets, in each simulation of a feedback loop, we use one of the recommendation strategies listed in Section 5.3.3 to recommend items to each user. After that, we simulate the users' response by assuming that they responded to items that are recommended and provide the true ratings for the top-N ratings (top-N=10). After new ratings are taken in, we update the recommender system. We then simulate runs of *Max Feedback Loops* = 20 iterations in Algorithm 1 where a single iteration (or loop) consists of the algorithm providing a recommendation, the user labeling the recommendation, and the algorithm updating its model of the user's preferences.

For each of the recommendation strategies in section 5.3.3, we set the number of

items (top-N) selected after each recommendation to 10. For the blind spot aware MF and conventional MF with and without AL models, we use stochastic gradient descent to optimize the objective function. For all the Matrix Factorization methods, we set the dimension of the latent space to 20. For all the four propensity based MF algorithms, we optimize the objective function following coordinate gradient descent update rules to learn the model [161]. For all gradient descent optimization updates, we set the number of learning iterations $steps = 200$, regularization weight $\lambda = 0.02$ and learning rate $\eta = 0.002$ for training the matrix factorization model. All the results are from the average of 10 independent runs.

We also report the results for random selection as baseline. This means that the items are selected randomly by the user to rate after the recommendations are made by using the Conventional MF model (meaning essentially an *open loop*).



(a) Training Errors

(b) Testing Errors

Figure 5.5: MAE and RMSE for training and testing for different β. The x-axis indicates different β, and the y-axis is the error. As shown here, a high β comes with a higher training error. Note that here, we recover pure matrix factorization with Frobenius normalization when $\beta = 0$ (see Eq. 5.7).

For the Blind Spot Aware MF, we need to decide what the weight for the blind spot aware term should be. We run experiments with different β on the original Movielens 100k data (before the completion), after randomly splitting the ratings into training (80%) and testing (20%). Based on Figure 5.5, we set β as 0.2 for the Blind Spot Aware MF, because the training error and testing error are relatively low with $\beta = 0.2$. The same procedure is

applied for the pure synthetic data, resulting in $\beta = 0.2$ as well.

5.4.4 Results for Synthetic Data

RMSE and MAE

Recording the RMSE and MAE helps us understand how the iterative recommender system is performing. Figure 5.6 shows the RMSE of each recommendation strategy during the iterative recommender system for the training ratings. The conventional MF and the conventional MF with AL show similar trends in the RMSE (identical on the plot). However, the conventional MF with random selection starts with a high RMSE, but this decreases with each iteration when more ratings are collected. All propensity based MF methods, with or without AL, show a similar trend for the training RMSE as well as the blind spot aware MF. At iteration 1, all six algorithms have the same initial training ratings. The reason why they have large differences on training RMSE lies in the fact that all the propensity MF strategies do not minimize the RMSE; instead they minimize the primary loss term consisting of the square error loss between the prediction $\hat{r}_{ui}$ and the true rating r_{ui}, inversely weighted by $\frac{1}{P_{u,i}}$ (see Eq. 5.1). On the other hand, conventional MF directly minimizes RMSE, i.e. does not weight this error in the primary loss term (which is equivalent to setting $P_{u,i} = 1$). Note that random baseline means that the items are selected randomly after the recommendations (meaning essentially an open loop). Figure 5.7 shows the MAE of the training set with iterations.

We also record the RMSE and MAE on the test ratings shown in Figure 5.8 and Figure 5.9. We can see that the RMSE for the test set increases dramatically in the early stage for propensity based MFs, but then drops to a low level. On the other hand, random selection and conventional MF approaches have a decreasing trend in RMSE with iterations, however with a higher value compared to other algorithms. The blind spot aware MF algorithm has the lowest testing error compared to other algorithms.

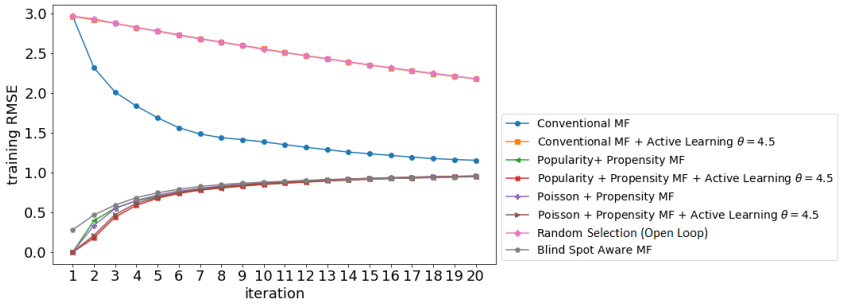


Figure 5.6: Training RMSE with different debiasing mechanisms for each iteration on pure synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the RMSE for the training set. The conventional MF and the conventional MF with AL show a similar trend during each iteration. All four propensity based MFs also show a similar trend as well as the blind spot aware MF. Note that random baseline means that the items are selected randomly after the recommendations (meaning essentially an open loop).

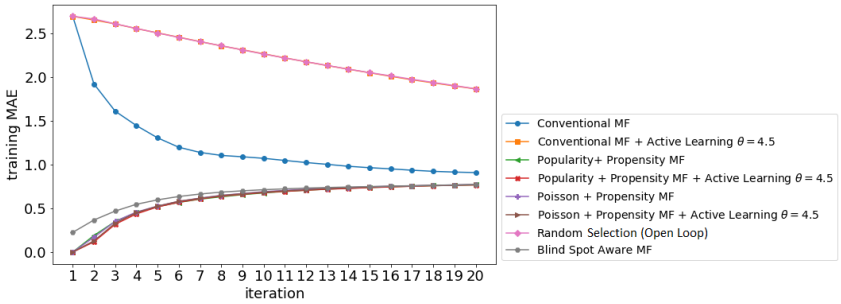


Figure 5.7: Training MAE with different debiasing mechanisms for each iteration on pure synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the MAE for the training set. The training MAE has a trend similar to the training RMSE trend (see figure 5.6).

Gini Coefficient

As stated in section 5.4, we computed the Gini coefficient of the item popularity after each feedback loop iteration to assess how different debiasing mechanisms affect the distribution of popularity. A higher Gini index indicates more heterogeneity of the popularity, essentially meaning a bigger popularity divide between the items, leading to a wider gap between the haves and the have-nots. Figure 5.10 shows the Gini coefficient distribution. Conventional MF increases the inequality of the popularity of items, which indicates that the conventional MF will boost some popular items. On the other hand, the pure Propensity based MF has a high Gini coefficient at an early stage which then decreases

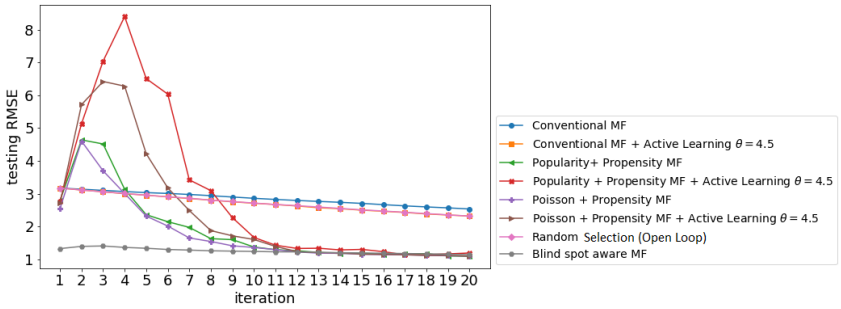


Figure 5.8: Testing RMSE with different debiasing mechanisms for each iteration on pure synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the RMSE for the testing set. The conventional MF with and without AL show a similar trend during each iteration, and they are similar to random selection. All four propensity based MFs also show a similar trend with lower RMSE, as well as the blind spot aware MF.

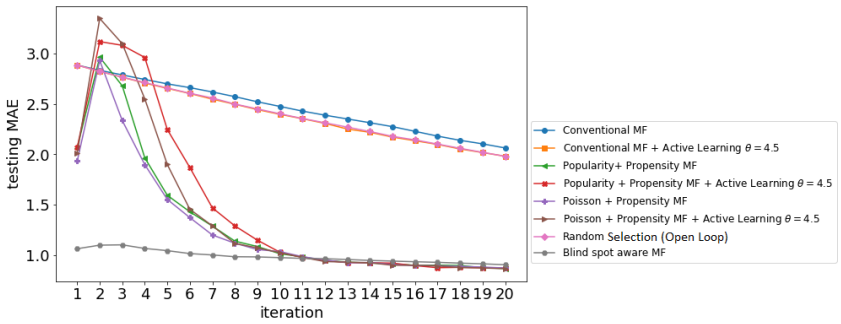


Figure 5.9: Testing MAE with different debiasing mechanisms for each iteration on pure synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the MAE for the testing set. The testing MAE has a trend similar to the training RMSE trend (see figure 5.6).

with each feedback loop iteration. Propensity based MF with AL results in a low Gini coefficient across all feedback loop iterations. Random selection appears to have the lowest Gini coefficient since all items have the same probability of being explored. Blind Spot Aware MF has a lower Gini coefficient compared with Conventional MF.

Blind Spot

As stated in section 5.4, we recorded the blind spot score for each item during each iteration with a cutoff $\epsilon = 0.9$. Figure 5.11 shows the blind spot score at time t_1 and t_{20} with all methods. With more ratings, all used methods decreased the blind spot score. However, the propensity based MF brought all items closer so that the blind spot score was more

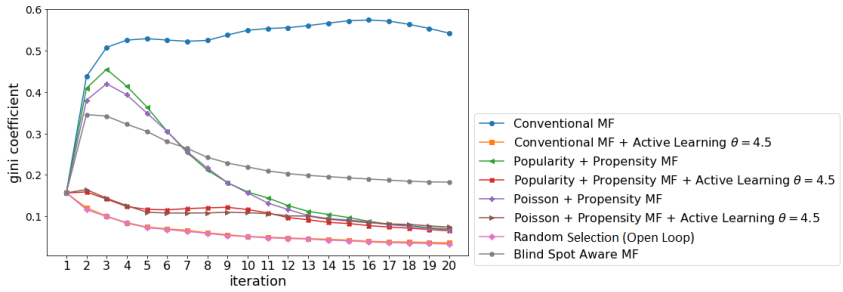
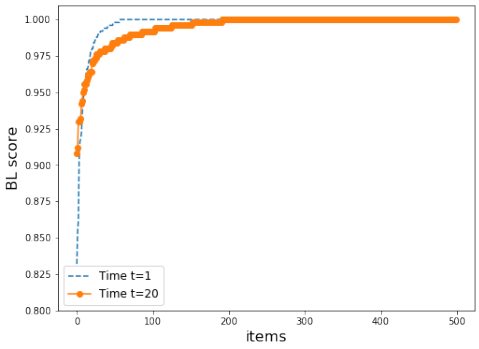
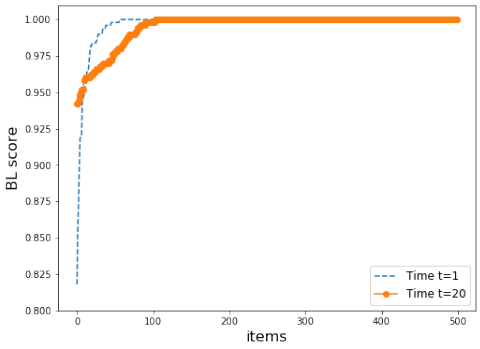


Figure 5.10: Gini coefficient vs. feedback loop iteration on the synthetic data. The Gini coefficient increases for the Popularity Propensity MF without AL, but then quickly decreases. On the other hand, the Gini coefficient score of the Propensity MF with AL continues to decrease. Note that a higher Gini index indicates more heterogeneity of the popularity, essentially meaning a bigger popularity divide between the items.

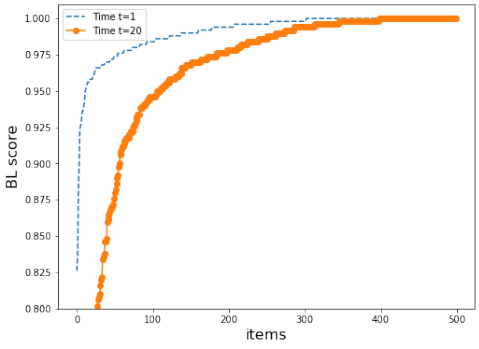
evenly distributed. Within each iteration, new items were added for the conventional MF as well (see sub-figures (a) and (b) in figure 5.11). This brought a certain amount of diversity to the item pool, and flattened the blind spot score distribution a little. At iteration $t = 20$, the two propensity-based MFs and Blind spot aware MF have significant difference compared to the conventional MF on the BL score, and the p_{value} of Mann-Whitney U test is 7.2e-10, 6.2e-10 and 2.2e-13 for popularity propensity MF, Poisson propensity MF and Blind spot aware MF separately. Figure 5.12 shows a more clear comparison between different algorithms at time $t = 20$. The conventional MF resulted in a ranking-based blind spot (items whose predicted ratings are below 90% of the maximum predicted ratings) ranging between 95% and 99% of all items on average. Both propensity-based MF methods resulted in blind spots consisting of between 94% and 96% of all items; while the blind spot aware MF resulted in a ranking-based blind spot with around 90% to 94% of all items.



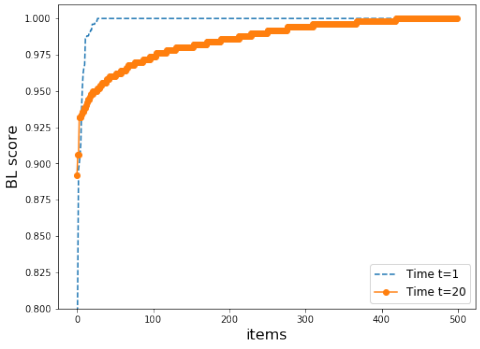
(a) BL score for conventional MF with random selection



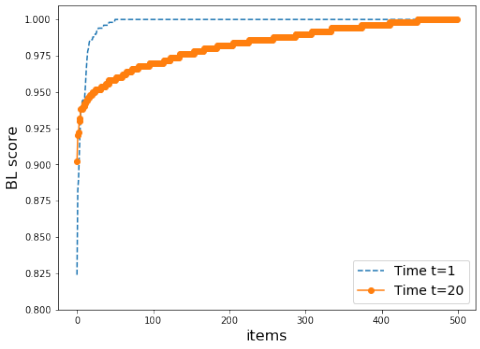
(b) BL score for conventional MF



(c) BL score for Blind spot aware MF



(d) BL score for popularity propensity MF



(e) BL score for popularity propensity MF

Figure 5.11: Blind spot score vs. iteration on pure synthetic data. The x-axis represents all the items sorted by the BL score in an ascending order, and the y-axis represents the BL score (see section 5.4). As shown, all four main algorithms help to flatten the blind spot score as more ratings are introduced. Both propensity based MFs have a significant effect on the BL score distribution. On the other hand, the blind spot aware MF also flattens the BL score. The random selection and the conventional MF flatten the BL score less than the other three algorithms.

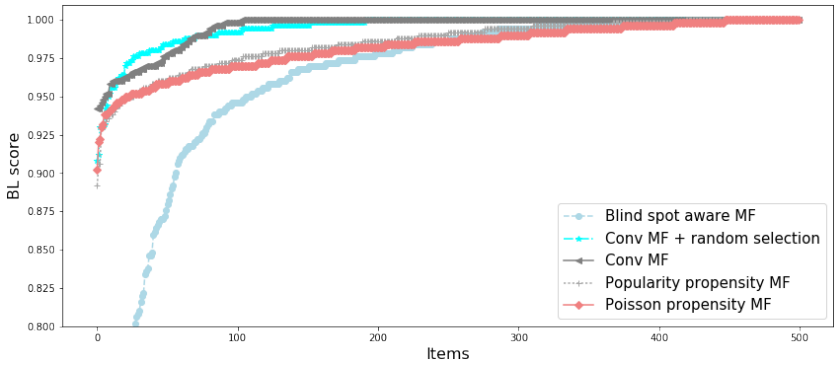


Figure 5.12: Blind spot score vs. different algorithms at time $t = 20$ on pure synthetic data. The x-axis represents all the items sorted by the BL score, and the y-axis represents the BL score (see section 5.4).

Varying θ for the active recommendation strategy

In order to check how θ affects the distribution of predictions for the algorithms in Section 5.3.3, we ran experiments with three conditions: $\theta = (3.5, 4, 4.5)$ and compared the results.

We fixed the recommendation strategy as the conventional matrix factorization and then we varied θ with an active learning strategy at each iteration (see Section 5.3.2). Figure 5.13 shows the Gini coefficient of popularity at each iteration. We found that the conventional matrix factorization technique significantly increased the Gini coefficient and that there was no significant difference between different θ. However, the active recommendation strategy successfully decreases the Gini coefficient to a level which is similar to random selection. All the three active recommendation strategies and Blind spot aware MF have significantly smaller Gini coefficient ($p_{value} < 0.01$). The reason is that the predicted ratings with conventional MF have high diversity, in the range (3.5, 4.5) compared to the maximum rating of 5.0.

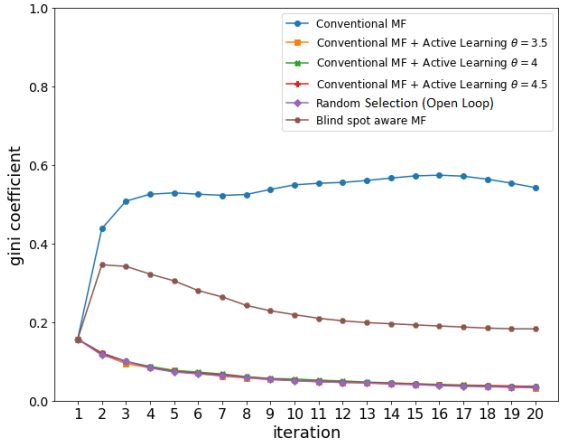


Figure 5.13: The x-axis represents the feedback loop iteration number, and the y-axis represents the Gini coefficient of popularity score at each iteration. As shown, the Gini coefficient increases for the conventional matrix factorization for all four different θ with the conventional MF. The blind spot aware MF has a certain level of balancing the popularity of items during iterative RS.

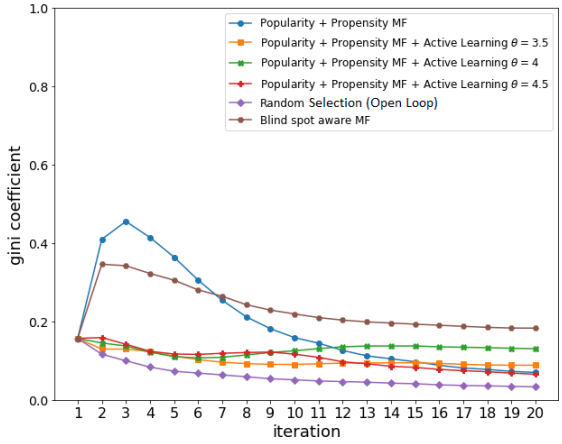


Figure 5.14: Gini coefficient vs. iteration with different θ when using the popularity propensity MF on pure synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the Gini coefficient of the popularity score in each iteration. As shown, the Gini coefficient shows no significant difference with different θ. The blind spot aware MF has a certain level of balancing the popularity of items.

We also recorded how different θ affect the Gini coefficient of the popularity propen-

sity MF and the Poisson propensity MF (see figure 5.14 and figure 5.15). Both propensity based MFs have a small Gini coefficient of popularity of items during each iteration. The reason that both propensity based MFs are not affected by different θ is that the cost function takes into account the propensity (or the probability that a user u sees an item i), which increases the chance of unpopular items to be explored by the recommender system.

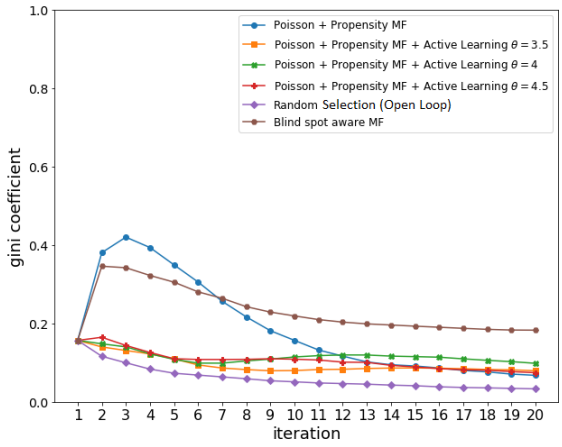
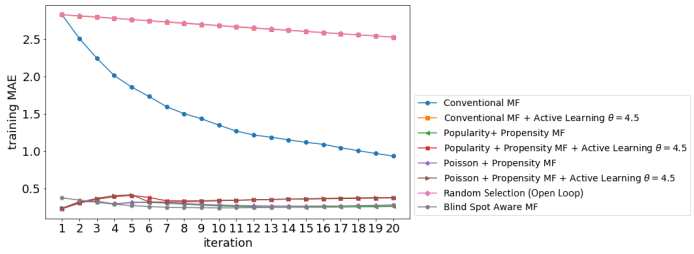


Figure 5.15: Gini coefficient vs. iteration with different θ when using the Poisson propensity MF on pure synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the Gini coefficient of popularity score at each iteration. As shown, when we vary θ, the Poisson propensity MF has a similar trend as the popularity propensity MF (see figure 5.14). Note that a higher Gini index indicates more heterogeneity of the popularity, essentially meaning a bigger popularity divide between the items.

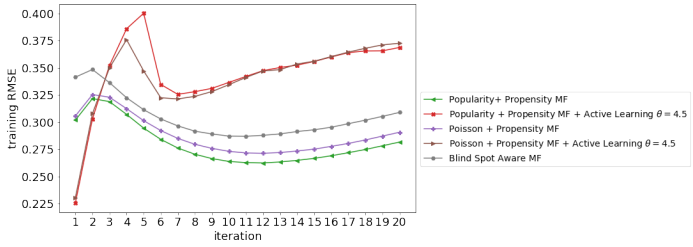
5.4.5 Results for semi-synthetic Data

In this section, we used a semi-synthetic dataset to demonstrate how different debiasing algorithms affect the distribution of popularity of items and the blind spot score for each iteration. As described in Section 5.4.1, the semi-synthetic dataset is obtained after completion of the original MovieLens 100k dataset.

RMSE and MAE



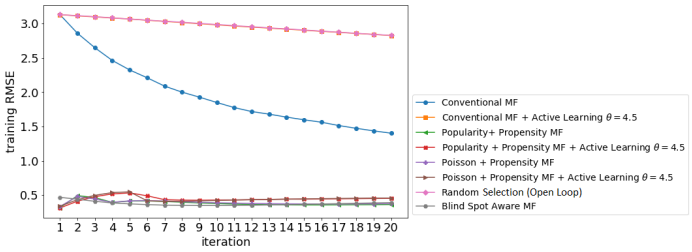
(a) All algorithms comparison



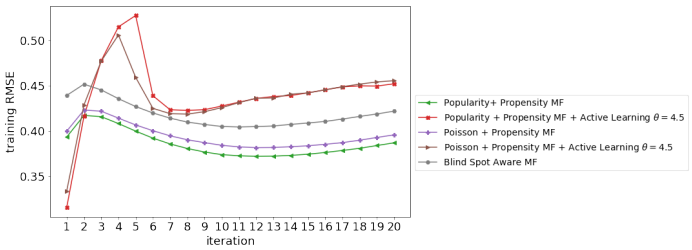
(b) Four propensity based MFs and blind spot aware MF

Figure 5.16: The training MAE with different debiasing mechanisms on semi-synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the MAE for the training data. The conventional MF and its combination with active learning have almost the same training MAE. After 20 iterations, the different algorithms tend to achieve good training MAE as more ratings become available for collaborative filtering, except the random selection algorithm. The sub-figure (b) is part of figure (a) for a clear comparison.

The overall MAE for the conventional MF with active learning shows the same trend as the conventional MF in Figure 5.16. As shown, MAE decreases with iterations which indicates two main phenomena: 1) the propensity MF has a better performance compared to the conventional MF; 2) the propensity based MF has a similar effect on the training MAE across all four strategies; 3) random selection only slightly decreases the training MAE as each iteration continues; and 4) the blind spot aware MF has the lowest training error.



(a) All algorithm comparison



(b) Four propensity based MFs and blind spot aware MF

Figure 5.17: The training RMSE with different debiasing mechanisms on semi-synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the RMSE for the training data. The sub-figure (b) is part of figure (a) for a clear comparison. As shown, the training RMSE shows a similar trend as the MAE (see figure 5.16).

As for the training RMSE, the conventional MF and the conventional MF with AL have similar trends. In contrast, random selection has a higher RMSE which decreases with each iteration (see figure 5.17). All propensity based MF methods, with or without AL, have a similar trend for the training RMSE. The blind spot aware MF has the lowest training error.

We also record RMSE and MAE on the testing ratings as shown in Figure 5.18 and Figure 5.19. We can see that the RMSE for the test set increases dramatically in the early stages before the 5th iteration, but then drops to a low level. On the other hand, random selection and conventional MF approaches have a decreasing trend on RMSE with more iterations. As shown here, the propensity MF algorithms and the blind spot aware MF algorithm have a lower testing error compared to other algorithms.

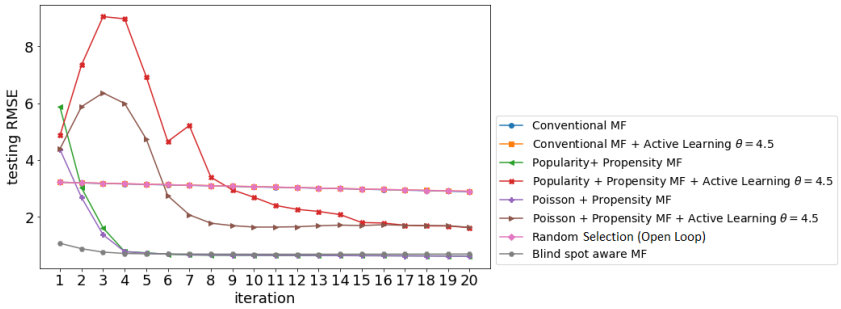


Figure 5.18: Testing RMSE with different debiasing mechanisms for each iteration on semi-synthetic data. The x-axis represents the iteration number, and the y-axis represents the RMSE for the testing set. The conventional MF with and without AL show a similar trend during each iteration, and are similar to random selection. All four propensity based MFs also show a similar trend with lower RMSE.

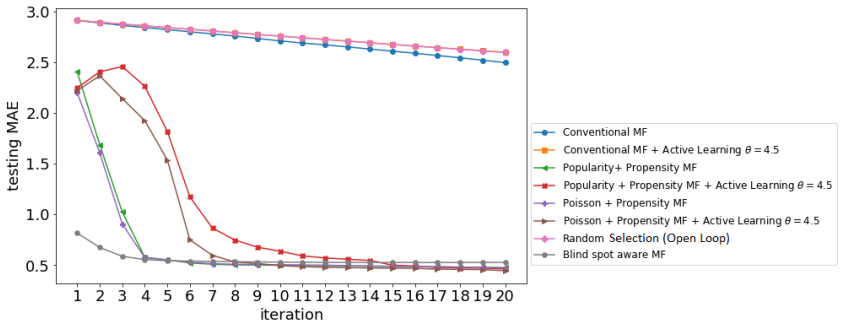


Figure 5.19: Testing MAE with different debiasing mechanisms for each iteration on semi-synthetic data. The x-axis represents the feedback loop iteration number, and the y-axis represents the MAE for the testing set. The testing MAE has a trend similar to the training RMSE trend (see Figure 5.18).

Gini Coefficient

Figure 5.20 shows that the Gini coefficient increases in the early stages for all algorithms. This is partly because the initial rating matrix is randomly selected and because all items have a similar popularity score, which means that they have a similar probability of being explored by users.

We can see from Figure 5.20 that all four propensity based MFs start with an increasing Gini index. The popularity propensity MF and Poisson propensity MF combined with AL have a lower Gini coefficient during all iterations. Although the blind spot aware MF increases the Gini coefficient, it is still less than that of the conventional matrix factorization. Random selection intuitively has the lowest Gini coefficient score through all iterations.

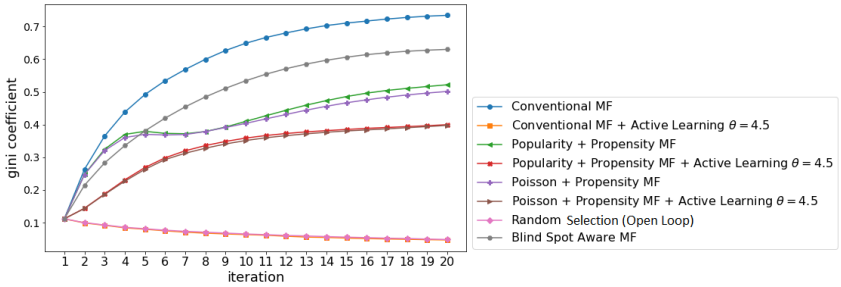


Figure 5.20: Gini coefficient vs. iteration with different debiasing mechanisms on semi-synthetic data. As shown, the Gini coefficient increases in the early stage for all mechanisms except random selection. The Gini index of all propensity based methods and their combination with active learning have a smaller Gini coefficient than that of the conventional MF ($p_{value} < 0.05$). The proposed blind spot aware MF has a certain level of debiasing compared to conventional MF ($p_{value} < 0.05$).

Varying θ for the active recommendation strategy

In order to check how θ affects the distribution of predictions for the algorithms in Section 5.3.3, we ran experiments with three conditions: $\theta = (3.5, 4, 4.5)$ and compared the results for the semi-synthetic dataset following the same procedure as in the synthetic dataset.

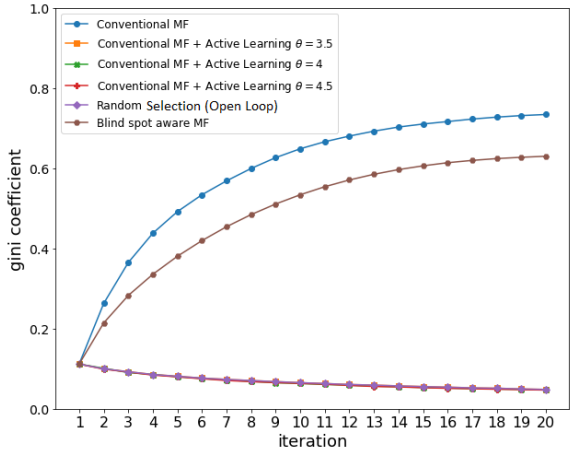


Figure 5.21: Gini coefficient vs. iteration with different θ on conventional matrix factorization on semi-synthetic data. As shown, the Gini coefficient increases for the conventional matrix factorization with different θ. There is no significant difference across all three θ.

We fixed the recommendation strategy as the conventional matrix factorization and then we varied θ with an active learning strategy at each iteration as explained in Section 5.3.2. Figure 5.21 shows the Gini coefficient of popularity at each iteration. We found that the conventional matrix factorization technique significantly increased the Gini coefficient and that there was no significant difference between different θ. All the AL-associated conventional MFs have low Gini coefficient value, similar to the pure synthetic data experiment results (see Figure 5.14). On the other hand, we found that random selection resulted in the smallest Gini coefficient.

Figure 5.22 shows how θ affects the Gini coefficient of the popularity with the popularity propensity matrix factorization. Without an active learning strategy, the popularity propensity MF increases the Gini coefficient dramatically at an early stage. Different θ have similar effects on the Gini coefficient, and it is generally smaller with lower θ. Our proposed blind spot aware MF has a higher Gini coefficient compared to the popularity propensity based MFs with and without AL strategy.

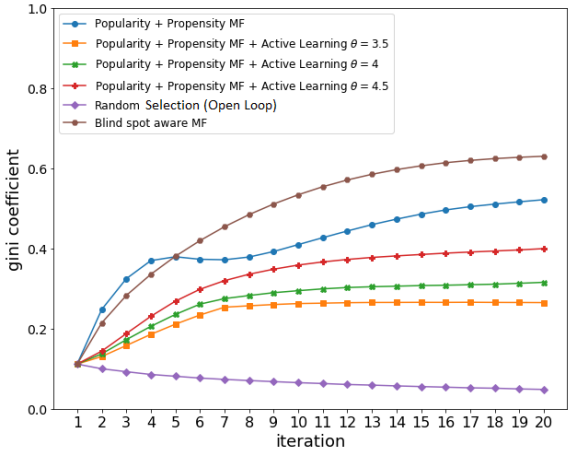


Figure 5.22: Gini coefficient vs. iteration with different θ on the popularity propensity matrix factorization on semi-synthetic data. As shown, the Gini coefficient increases for the popularity propensity matrix factorization. The Gini coefficient with $\theta = 3.5$ has the smallest Gini coefficient. On the other hand, the blind spot aware MF has a similar Gini coefficient trend as the popularity propensity MF.

Figure 5.23 shows how θ affects the Gini coefficient of popularity with the Poisson propensity MF, which has a similar trend as the popularity propensity MF.

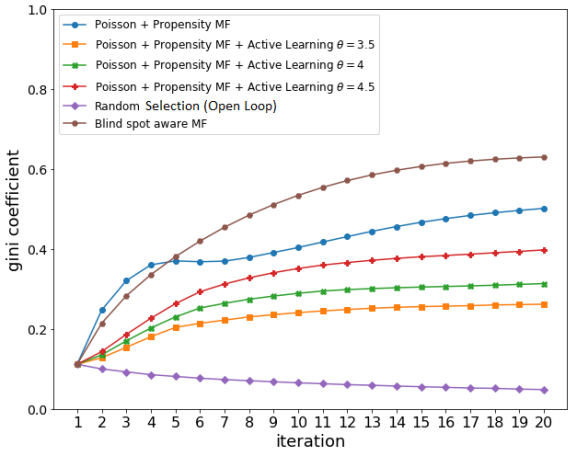
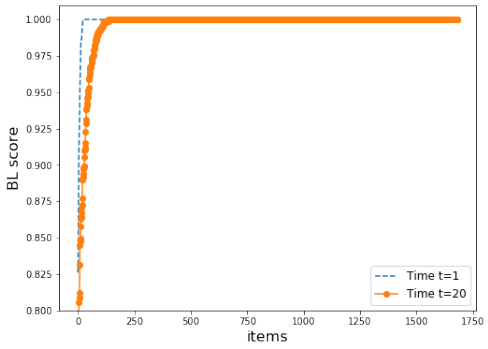


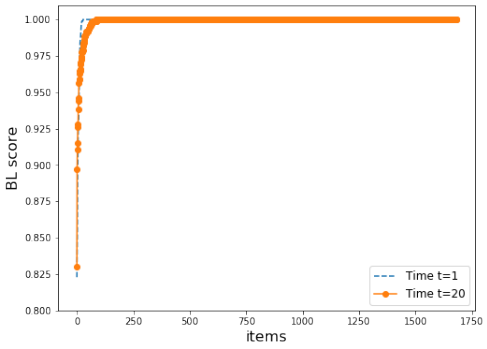
Figure 5.23: Gini coefficient vs. iteration with different θ on Poisson propensity matrix factorization on semi-synthetic data. As shown, the Gini coefficient shows similar trends with Figure 5.22, which confirms that propensity matrix factorization leads to the low Gini coefficient.

Blind spot analysis

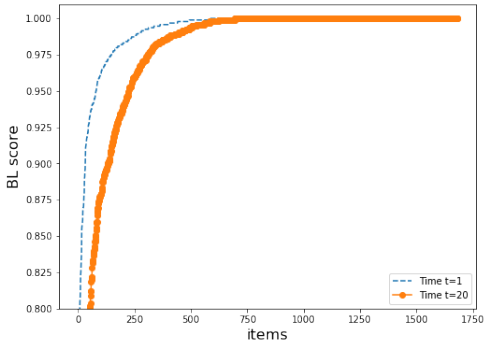
As stated in Section 5.4, we also recorded the BL score on our semi-synthetic dataset with a cutoff $\epsilon = 0.9$. As shown in figure 5.24, Propensity-based MFs and Blind spot aware algorithms help to flatten the blind spot score as more ratings are introduced. Both propensity based MFs have a significant effect on the BL score distribution, even from the early stage. The conventional MF with random selection has a small effect on the BL score distribution. Although random selection provides us with a large advantage to discover more items, it does not show much improvement on the blind spot score here. At iteration $t = 20$, the two propensity-based MFs and Blind spot aware MF have significant difference compared to conventional MF on the BL score, and the p_{value} of Mann-Whitney U test is 9e-10, 8.2e-10 and 2e-15 for popularity propensity MF, Poisson propensity MF and Blind spot aware MF. Figure 5.25 shows a more clear comparison between different algorithms at time $t = 20$. Conventional matrix factorization using 20 latent factors, resulted in a ranking-based blind spot containing between 95% and 99% of all items. Popularity-based and Poisson based propensity-based MFs resulted in a ranking-based blind spot with between 96% and 97% if all items; while the blind spot aware MF resulted in a ranking-based blind spot with between 92% and 96% if all items.



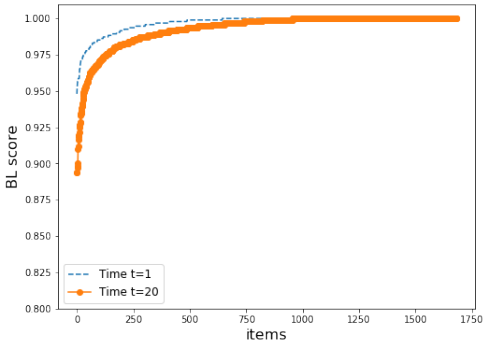
(a) BL score for conventional MF with random selection



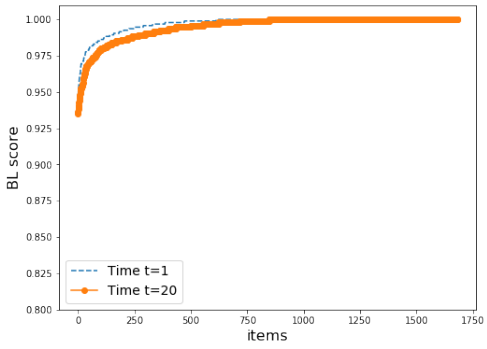
(b) BL score for conventional MF



(c) BL score for Blind spot aware MF



(d) BL score for popularity propensity MF



(e) BL score for popularity propensity MF

Figure 5.24: Blind spot score vs. iteration on semi-synthetic data. The x-axis represents all the items sorted by the BL score in an ascending order, and the y-axis represents the blind spot score (see Section 5.4). As shown, the propensity based MFs and blind spot aware algorithms help to flatten the blind spot score at the early stage. Both propensity based MFs have a significant effect on the BL score distribution. The conventional MF with random selection has a small effect on the BL score distribution. The conventional MF has no significant effect on the BL score.

Based on the observed trends of the MAE/RMSE, the Gini coefficient, and the blind spot analysis, we conclude that: 1) the propensity MF achieves a significant level of debiasing on the recommendations in terms of Gini coefficient ($p_{value} < 0.05$); 2) active learning combined with propensity MF results in low MAE and a high debiasing effect on recommendations; 3) conventional matrix factorization increases the bias which eventually affects the ability to discover new items, and decreases the bias when combined with an active learning strategy; and 4) the blind spot aware MF result in a certain level of debiasing RS in terms of the blind spot score and Gini coefficient.

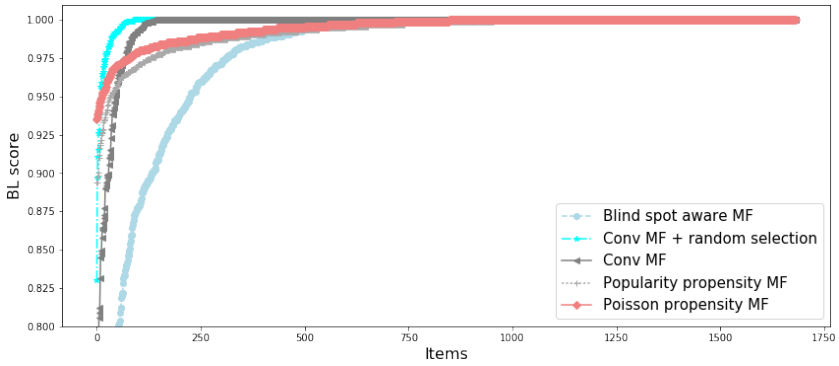


Figure 5.25: Blind spot score vs. different algorithms at time $t = 20$ with semi-synthetic data. The x-axis represents all the items sorted by the BL score, and the y-axis represents the BL score (see section 5.4).

5.5 Summary and Conclusions

Recommender systems introduce bias during the interactive feedback loop with users over time, which might cause them to make more biased recommendations. In this chapter, we introduced an interactive framework to simulate the feedback loop that is created by the chain of events generated when a recommender system interacts with users over time. Based on this framework, we proposed several debiasing algorithms based on existing techniques to use during this chain of events. We also proposed a blind spot aware matrix factorization which takes into account the blind spot score when trying to learn the recommendation model. Note that in this chapter, we do not focus on improving collaborative filtering algorithms (Matrix Factorization) for recommender systems by studying user feedback.

Instead, our goal is to simulate the interaction between users and the recommender system and to debias the recommender system during the interaction. Our results showed that the propensity MF achieved a certain level of debiasing of the RS while active learning combined with the propensity MF achieved a higher debiasing effect on recommendations. Our proposed blind spot aware matrix factorization also achieved a certain level of debiasing of the RS. One limitation of this study is that we assume that users totally agree with the recommendations in each iteration, and provide feedback. In real-life, users might not agree with recommendations. But we aim to study the effect of interaction itself, thus future study could introduce more realistic user reactions as we have done for the content-based filtering in [152], and in chapter 4 (RQ 3 and RQ 4). Another limitation is the lack of applicable data. As noted by Shafto and Nasraoui in [77,130], most data sets in RS are static, they do not consider the iterative effect of the interaction between users and RS. Therefore, future work could perform a user study to collect data considering the iterated interaction effect.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

In this dissertation, we first investigated three forms of iterated algorithmic bias (filter, active learning, and a random baseline) and how they affect the performance of machine learning algorithms by formulating research questions about the impact of each type of bias. Based on statistical analysis of the results of several controlled experiments using synthetic and real data, we found that:

1. The three different forms of iterated algorithmic bias (filter, active learning, and random selection, used as query mechanisms to show data and request new feedback/labels from the user), **do affect algorithm performance** when fixing the human interaction probability to 1.

2. Different initial class imbalance in the training data used to generate the initial relevance boundary, significantly affect the machine learning algorithm's results for all three forms of iterated algorithmic bias, **impacting boundary shift, heterogeneity of predicted relevance, and hidden relevant items (class 1-blind spot)**.

3. Iterated filter bias has a more significant effect on the class-1-blind spot size compared to the other two forms of algorithmic biases. **This means that iterated filter bias, which is prominent in personalized user interfaces, can limit humans' ability to discover data that is relevant to them**.

4. The iterated learning framework is effective for analyzing the impact of iterated algorithmic bias in human-algorithm interaction.

Future work can consider the following three directions: 1) Taking into account additional parameters and configurations when testing the impact of iterated algorithmic bias and hu-

man interaction on ML models, including: (A) Changing the number of items recommended in top N relevant item recommendation lists and (B) Applying different types of AL (we only investigated uncertainty-based AL); 2) Human experiments that study research questions that are similar to the ones formulated for the simulated experiments; 3) In our study, we considered three iterated algorithmic biases modes, there are more possible iterated algorithmic biases modes, such as active-filter bias (Combining active learning and filter bias, i.e., querying data which are close to the center of each class); 4) In our work, we used two different models to model human 4action, but human reaction is more complicated. Future work would be to develop more realistic human reaction models.

Secondly, we investigated one major problem of recommender systems, iterated bias that builds up during its continuous feedback loop with users over time. We viewed a RS environment as generating a continuous chain of events as a result of the interactions between users and the recommender system outputs over time. Based on this model, we proposed several debiasing algorithms during this chain of events using existing techniques and evaluated how they impact prediction accuracy as well as the trends of increase or decrease in the inequality of the popularity distribution of items over time. We also proposed a novel blind spot aware matrix factorization to debias the recommender system. Results showed that the propensity MF achieved a certain level of debiasing of the RS, while active learning combined with the propensity MF achieved a higher debiasing effect on recommendations. Our proposed blind spot aware matrix factorization also achieved a certain level of debiasing of the RS.

One limitation of this study is that we assume that users totally agree with the recommendations in each iteration, and provide feedback. In real-life, users might not agree with recommendations. Future study could introduce more realistic user reaction models. Another limitation is the lack of applicable data. Most data sets used in evaluating RSs are static, they do not consider the iterative effect of the interaction between users and RS. Therefore, future studies could design user experiments to collect data considering the iterated interaction effect.

There is a long tradition in machine learning of algorithms whose performance is guaranteed in the context of unbiased data. Similarly, there is a long tradition in the psychology of human learning of treating learning as inference from unbiased data. Increasingly, people and algorithms are engaged in interactive processes wherein neither the humans nor the algorithms receive unbiased data. Our research attempted to better understand how algorithm performance and human feedback or data depend on one another and how those dependencies affect long run performance. Our ongoing work will pave the road for a framework in which the study of human-algorithm interaction may progress.

REFERENCES

[1] Karen Sparck Jones, "Some thoughts on classification for retrieval," *Journal of Documentation*, vol. 26, no. 2, pp. 89–101, 1970.

[2] Stephen E Robertson, "The probability ranking principle in ir," *Journal of documentation*, vol. 33, no. 4, pp. 294–304, 1977.

[3] K Jones Spark, "Artificial intelligence: What can it offer to information retrieval," *Proceedings of the Informatics 3, Aslib, ed., London*, 1978.

[4] Cornelis Joost Van Rijsbergen, "Information retrieval," 1979.

[5] Gerard Salton, Edward A Fox, and Harry Wu, "Extended boolean information retrieval," *Communications of the ACM*, vol. 26, no. 11, pp. 1022–1036, 1983.

[6] Bruce Croft, Donald Metzler, and Trevor Strohman, "Search engines: Information retrieval in practice, 2008," .

[7] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al., *Modern information retrieval*, vol. 463, ACM press New York, 1999.

[8] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. 35, no. 12, pp. 61–70, 1992.

[9] Pattie Maes et al., "Agents that reduce work and information overload," *Communications of the ACM*, vol. 37, no. 7, pp. 30–40, 1994.

[10] Michael Pazzani and Daniel Billsus, "Learning and revising user profiles: The identification of interesting web sites," *Machine learning*, vol. 27, no. 3, pp. 313–331, 1997.

[11] Paul Resnick and Hal R Varian, "Recommender systems," *Communications of the ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[12] Chumki Basu, Haym Hirsh, William Cohen, et al., "Recommendation as classification: Using social and content-based information in recommendation," in *Aaai/iaai*, 1998, pp. 714–720.

[13] J Ben Schafer, Joseph Konstan, and John Riedl, "Recommender systems in e-commerce," in *Proceedings of the 1st ACM conference on Electronic commerce*. ACM, 1999, pp. 158–166.

[14] Gediminas Adomavicius and Alexander Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE transactions on knowledge and data engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[15] Olfa Nasraoui and Mrudula Pavuluri, "Complete this puzzle: a connectionist approach to accurate web recommendations based on a committee of predictors," in *International Workshop on Knowledge Discovery on the Web*. Springer, 2004, pp. 56–72.

[16] Behnoush Abdollahi and Olfa Nasraoui, "A cross-modal warm-up solution for the cold-start problem in collaborative filtering recommender systems," in *Proceedings of the 2014 ACM conference on Web science*. ACM, 2014, pp. 257–258.

[17] Behnoush Abdollahi, "Accurate and justifiable: new algorithms for explainable recommendations.," 2017.

[18] Behnoush Abdollahi and Olfa Nasraoui, "Transparency in fair machine learning: the case of explainable recommender systems," in *Human and Machine Learning*, pp. 21–35. Springer, 2018.

[19] Behnoush Abdollahi and Olfa Nasraoui, "Using explainability for constrained matrix factorization," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017, pp. 79–83.

[20] Behnoush Abdollahi and Olfa Nasraoui, "Explainable restricted boltzmann machines for collaborative filtering," *arXiv preprint arXiv:1606.07129*, 2016.

[21] Michael Pazzani and Daniel Billsus, "Content-based recommendation systems," *The adaptive web*, pp. 325–341, 2007.

[22] Marko Balabanović and Yoav Shoham, "Fab: content-based, collaborative recommendation," *Communications of the ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[23] Upendra Shardanand and Pattie Maes, "Social information filtering: algorithms for automating word of mouth," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co., 1995, pp. 210–217.

[24] Joseph A Konstan, Bradley N Miller, David Maltz, Jonathan L Herlocker, Lee R Gordon, and John Riedl, "Grouplens: applying collaborative filtering to usenet news," *Communications of the ACM*, vol. 40, no. 3, pp. 77–87, 1997.

[25] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*. ACM, 2001, pp. 285–295.

[26] Greg Linden, Brent Smith, and Jeremy York, "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no. 1, pp. 76–80, 2003.

[27] Thomas Cover and Peter Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[28] Sahibsingh A Dudani, "The distance-weighted k-nearest-neighbor rule," *IEEE Transactions on Systems, Man, and Cybernetics*, , no. 4, pp. 325–327, 1976.

[29] Daniel D Lee and H Sebastian Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[30] Yehuda Koren, Robert Bell, and Chris Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, 2009.

[31] Ruslan Salakhutdinov and Andriy Mnih, "Probabilistic matrix factorization.," in *Nips*, 2007, vol. 1, pp. 2–1.

[32] Douglas L Medin and Marguerite M Schaffer, "Context theory of classification learning.," *Psychological review*, vol. 85, no. 3, pp. 207, 1978.

[33] Robert M Nosofsky, "Choice, similarity, and the context theory of classification.," *Journal of Experimental Psychology: Learning, memory, and cognition*, vol. 10, no. 1, pp. 104, 1984.

[34] William H Kruskal and W Allen Wallis, "Use of ranks in one-criterion variance analysis," *Journal of the American statistical Association*, vol. 47, no. 260, pp. 583–621, 1952.

[35] Thomas L Griffiths and Mark Steyvers, "Finding scientific topics," *Proceedings of the National academy of Sciences*, vol. 101, no. suppl 1, pp. 5228–5235, 2004.

[36] Thomas L Griffiths, Mark Steyvers, and Joshua B Tenenbaum, "Topics in semantic representation.," *Psychological review*, vol. 114, no. 2, pp. 211, 2007.

[37] Beerud Sheth and Pattie Maes, "Evolving agents for personalized information filtering," in *Artificial Intelligence for Applications, 1993. Proceedings., Ninth Conference on*. IEEE, 1993, pp. 345–352.

[38] Uri Hanani, Bracha Shapira, and Peretz Shoval, "Information filtering: Overview of issues, research and systems," *User modeling and user-adapted interaction*, vol. 11, no. 3, pp. 203–259, 2001.

[39] Joseph John Rocchio, "Relevance feedback in information retrieval," 1971.

[40] C Buckeley, G Salton, J Allan, and A Stinghal, "Automatic query expansion using smart," in *Proceedings of the 3rd Text Retrieval Conference*, 1994, pp. 69–80.

[41] Daniel Billsus and Michael J Pazzani, "Adaptive news access, the adaptive web: methods and strategies of web personalization," 2007.

[42] Olfa Nasraoui, Maha Soliman, Esin Saka, Antonio Badia, and Richard Germain, "A web usage mining framework for mining evolving user profiles in dynamic web sites," *IEEE transactions on knowledge and data engineering*, vol. 20, no. 2, pp. 202–215, 2008.

[43] Mohamed Koutheaïr Khribi, Mohamed Jemni, and Olfa Nasraoui, "Automatic recommendations for e-learning personalization based on web usage mining techniques and information retrieval," in *Advanced Learning Technologies, 2008. ICALT'08. Eighth IEEE International Conference on*. IEEE, 2008, pp. 241–245.

[44] Mohamed Koutheaïr Khribi, Mohamed Jemni, and Olfa Nasraoui, "Automatic personalization in e-learning based on recommendation systems: An overview," in *Intelligent and Adaptive Learning Systems: Technology Enhanced Support for Learners and Teachers*, pp. 19–33. IGI Global, 2012.

[45] Leyla Zhuhadar and Olfa Nasraoui, "A hybrid recommender system guided by semantic user profiles for search in the e-learning domain," *Journal of Emerging Technologies in Web Intelligence*, vol. 2, no. 4, pp. 272–281, 2010.

[46] Zhiyong Zhang and Olfa Nasraoui, "Mining search engine query logs for query recommendation," in *Proceedings of the 15th international conference on World Wide Web*. ACM, 2006, pp. 1039–1040.

[47] R Duncan Luce, *Individual choice behavior: A theoretical analysis*, Courier Corporation, 2005.

[48] Amos Tversky, "Elimination by aspects: A theory of choice.," *Psychological review*, vol. 79, no. 4, pp. 281, 1972.

[49] Daniel L McFadden, "Quantal choice analaysis: A survey," in *Annals of Economic and Social Measurement, Volume 5, number 4*, pp. 363–390. NBER, 1976.

[50] Patrick Shafto, Noah D Goodman, and Michael C Frank, "Learning from others the consequences of psychological reasoning for human learning," *Perspectives on Psychological Science*, vol. 7, no. 4, pp. 341–351, 2012.

[51] Patrick Shafto and Noah Goodman, "Teaching games: Statistical sampling assumptions for learning in pedagogical situations," in *Proceedings of the 30th annual conference of the Cognitive Science Society*. Cognitive Science Society Austin, TX, 2008, pp. 1632–1637.

[52] Elizabeth Bonawitz, Patrick Shafto, Hyowon Gweon, Noah D Goodman, Elizabeth Spelke, and Laura Schulz, "The double-edged sword of pedagogy: Instruction limits spontaneous exploration and discovery," *Cognition*, vol. 120, no. 3, pp. 322–330, 2011.

[53] Daphna Buchsbaum, Alison Gopnik, Thomas L Griffiths, and Patrick Shafto, "Children's imitation of causal action sequences is influenced by statistical and pedagogical evidence," *Cognition*, vol. 120, no. 3, pp. 331–340, 2011.

[54] Patrick Shafto, Noah D Goodman, and Thomas L Griffiths, "A rational account of pedagogical reasoning: Teaching by, and learning from, examples," *Cognitive psychology*, vol. 71, pp. 55–89, 2014.

[55] Russell Warner, Todd Stoess, and Patrick Shafto, "Reasoning in teaching and misleading situations.," in *CogSci*, 2011.

[56] Patrick Shafto, Baxter Eaves, Daniel J Navarro, and Amy Perfors, "Epistemic trust: Modeling children's reasoning about others' knowledge and intent," *Developmental Science*, vol. 15, no. 3, pp. 436–447, 2012.

[57] BS Eaves and Patrick Shafto, "Unifying pedagogical reasoning and epistemic trust," *Advances in child development and behavior*, vol. 43, pp. 295–319, 2012.

[58] Asheley R Landrum, Baxter S Eaves, and Patrick Shafto, "Learning to trust and trusting to learn: A theoretical framework," *Trends in Cognitive Sciences*, vol. 19, no. 3, pp. 109–111, 2015.

[59] Thomas L Griffiths and Michael L Kalish, "A bayesian view of language evolution by iterated learning," in *Proceedings of the Cognitive Science Society*, 2005, vol. 27.

[60] Simon Kirby, Mike Dowman, and Thomas L Griffiths, "Innateness and culture in the evolution of language," *Proceedings of the National Academy of Sciences*, vol. 104, no. 12, pp. 5241–5245, 2007.

[61] Aaron Beppu and Thomas L Griffiths, "Iterated learning and the cultural ratchet," in *Proceedings of the 31st annual conference of the cognitive science society*. Citeseer, 2009, pp. 2089–2094.

[62] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender systems handbook*, pp. 73–105. Springer, 2011.

[63] Robin Burke, "Hybrid recommender systems: Survey and experiments," *User modeling and user-adapted interaction*, vol. 12, no. 4, pp. 331–370, 2002.

[64] P Kazienko and M Kiewra, "Personalized recommendation of web pages," *Chapter*, vol. 10, pp. 163–183, 2004.

[65] J Ben Schafer, Joseph A Konstan, and John Riedl, "E-commerce recommendation applications," in *Applications of Data Mining to Electronic Commerce*, pp. 115–153. Springer, 2001.

[66] Huseyin Polat and Wenliang Du, "Svd-based collaborative filtering with privacy," in *Proceedings of the 2005 ACM symposium on Applied computing*. ACM, 2005, pp. 791–795.

[67] Ruslan Salakhutdinov and Andriy Mnih, "Probabilistic matrix factorization," Citeseer, 2011.

[68] Olfa Nasraoui and Chris Petenes, "An intelligent web recommendation engine based on fuzzy approximate reasoning," in *Fuzzy Systems, 2003. FUZZ'03. The 12th IEEE International Conference on*. IEEE, 2003, vol. 2, pp. 1116–1121.

[69] Paritosh Nagarnaik and A Thomas, "Survey on recommendation system methods," in *Electronics and Communication Systems (ICECS), 2015 2nd International Conference on*. IEEE, 2015, pp. 1496–1501.

[70] Gerard Salton, "The smart retrieval system experiments in automatic document processing," 1971.

[71] Raymond J Mooney and Loriene Roy, "Content-based book recommending using learning for text categorization," in *Proceedings of the fifth ACM conference on Digital libraries*. ACM, 2000, pp. 195–204.

[72] Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes, and Matthew Sartin, "Combining content-based and collaborative filters in an online newspaper," in *Proceedings of ACM SIGIR workshop on recommender systems*. Citeseer, 1999, vol. 60.

[73] Barry Smyth and Paul Cotter, "A personalised tv listings service for the digital tv age," *Knowledge-Based Systems*, vol. 13, no. 2, pp. 53–59, 2000.

[74] Marko Balabanović, "An adaptive web page recommendation service," in *Proceedings of the first international conference on Autonomous agents*. ACM, 1997, pp. 378–385.

[75] Tien T Nguyen, Pik-Mai Hui, F Maxwell Harper, Loren Terveen, and Joseph A Konstan, "Exploring the filter bubble: the effect of using recommender systems on content diversity," in *Proceedings of the 23rd international conference on World wide web*. ACM, 2014, pp. 677–686.

[76] Gediminas Adomavicius, Zan Huang, and Alexander Tuzhilin, "Personalization and recommender systems," *State-of-the-art decision making tools in the information-intensive age*, pp. 55–100, 2008.

[77] Olfa Nasraoui and Patrick Shafto, "Human-algorithm interaction biases in the big data cycle: A markov chain iterated learning framework," *arXiv preprint arXiv:1608.07895*, 2016.

[78] David A Cohn, Zoubin Ghahramani, and Michael I Jordan, "Active learning with statistical models," *Journal of artificial intelligence research*, vol. 4, no. 1, pp. 129–145, 1996.

[79] Daniel Joseph Hsu, "Algorithms for active learning," 2010.

[80] David D Lewis and William A Gale, "A sequential algorithm for training text classifiers," in *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*. Springer-Verlag New York, Inc., 1994, pp. 3–12.

[81] H Sebastian Seung, Manfred Opper, and Haim Sompolinsky, "Query by committee," in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992, pp. 287–294.

[82] Burr Settles, "Active learning literature survey," *University of Wisconsin, Madison*, vol. 52, no. 55-66, pp. 11, 2010.

[83] Andrew I Schein and Lyle H Ungar, "Active learning for logistic regression: an evaluation," *Machine Learning*, vol. 68, no. 3, pp. 235–265, 2007.

[84] Simon Tong and Daphne Koller, "Support vector machine active learning with applications to text classification," *The Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2002.

[85] Steven CH Hoi, Rong Jin, and Michael R Lyu, "Large-scale text categorization by batch mode active learning," in *Proceedings of the 15th international conference on World Wide Web*. ACM, 2006, pp. 633–642.

[86] Burr Settles and Mark Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2008, pp. 1070–1079.

[87] Cha Zhang and Tsuhan Chen, "An active learning framework for content-based information retrieval," *Multimedia, IEEE Transactions on*, vol. 4, no. 2, pp. 260–268, 2002.

[88] Gokhan Tur, Dilek Hakkani-Tür, and Robert E Schapire, "Combining active and semi-supervised learning for spoken language understanding," *Speech Communication*, vol. 45, no. 2, pp. 171–186, 2005.

[89] Neil Rubens, Mehdi Elahi, Masashi Sugiyama, and Dain Kaplan, "Active learning in recommender systems," in *Recommender systems handbook*, pp. 809–846. Springer, 2015.

[90] Simon Kirby, Tom Griffiths, and Kenny Smith, "Iterated learning and the evolution of language," *Current opinion in neurobiology*, vol. 28, pp. 108–114, 2014.

[91] Simon Kirby, Hannah Cornish, and Kenny Smith, "Cumulative cultural evolution in the laboratory: An experimental approach to the origins of structure in human language," *Proceedings of the National Academy of Sciences*, vol. 105, no. 31, pp. 10681–10686, 2008.

[92] Michael L Kalish, Thomas L Griffiths, and Stephan Lewandowsky, "Iterated learning: Intergenerational knowledge transmission reveals inductive biases," *Psychonomic Bulletin & Review*, vol. 14, no. 2, pp. 288–294, 2007.

[93] Kenny Smith, "Iterated learning in populations of bayesian agents," in *Proceedings of the 31st annual conference of the cognitive science society*. Citeseer, 2009, pp. 697–702.

[94] Kenny Smith, Simon Kirby, and Henry Brighton, "Iterated learning: A framework for the emergence of language," *Artificial life*, vol. 9, no. 4, pp. 371–386, 2003.

[95] Anna N Rafferty, Thomas L Griffiths, and Dan Klein, "Convergence bounds for language evolution by iterated learning," in *Proceedings of the Thirty-First Annual Conference of the Cognitive Science Society*, 2009.

[96] Amy Perfors and Daniel J Navarro, "Language evolution can be shaped by the structure of the world," *Cognitive science*, vol. 38, no. 4, pp. 775–793, 2014.

[97] Frank Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, pp. 386, 1958.

[98] Shai Shalev-Shwartz and Yoram Singer, "Online learning: Theory, algorithms, and applications," 2007.

[99] Tom M Mitchell, *The need for biases in learning generalizations*, Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ. New Jersey, 1980.

[100] Diana F Gordon and Marie Desjardins, "Evaluation and selection of biases in machine learning," *Machine learning*, vol. 20, no. 1-2, pp. 5–22, 1995.

[101] Raymond J Mooney, "Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning," *arXiv preprint cmp-lg/9612001*, 1996.

[102] David Danks and Alex John London, "Algorithmic bias in autonomous systems," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 2017, pp. 4691–4697.

[103] James J Heckman, "Sample selection bias as a specification error (with an application to the estimation of labor supply functions)," 1977.

[104] Charles Elkan, "The foundations of cost-sensitive learning," in *International joint conference on artificial intelligence*. Lawrence Erlbaum Associates Ltd, 2001, vol. 17, pp. 973–978.

[105] Bianca Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 114.

[106] Bianca Zadrozny, John Langford, and Naoki Abe, "Cost-sensitive learning by cost-proportionate example weighting," in *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*. IEEE, 2003, pp. 435–442.

[107] Miroslav Dudík, Steven J Phillips, and Robert E Schapire, "Correcting sample selection bias in maximum entropy density estimation," in *Advances in neural information processing systems*, 2006, pp. 323–330.

[108] Eyal Beigman and Beata Beigman Klebanov, "Learning with annotation noise," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*. Association for Computational Linguistics, 2009, pp. 280–287.

[109] Hamid Izadinia, Bryan C Russell, Ali Farhadi, Matthew D Hoffman, and Aaron Hertzmann, "Deep classifiers from image tags in the wild," in *Proceedings of the 2015 Workshop on Community-Organized Multimodal Mining: Opportunities for Novel Solutions*. ACM, 2015, pp. 13–18.

[110] Naresh Manwani and PS Sastry, "Noise tolerance under risk minimization," *IEEE transactions on cybernetics*, vol. 43, no. 3, pp. 1146–1151, 2013.

[111] Ricardo Baeza-Yates, "Data and algorithmic bias in the web," in *Proceedings of the 8th ACM Conference on Web Science*. ACM, 2016, pp. 1–1.

[112] Dino Pedreshi, Salvatore Ruggieri, and Franco Turini, "Discrimination-aware data mining," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 560–568.

[113] Keith Kirkpatrick, "Battling algorithmic bias: how do we ensure algorithms treat us fairly?," *Communications of the ACM*, vol. 59, no. 10, pp. 16–17, 2016.

[114] Kate Crawford, Mary L Gray, and Kate Miltner, "Big data— critiquing big data: Politics, ethics, epistemology— special section introduction," *International Journal of Communication*, vol. 8, pp. 10, 2014.

[115] Megan Garcia, "Racist in the machine: The disturbing implications of algorithmic bias," *World Policy Journal*, vol. 33, no. 4, pp. 111–117, 2016.

[116] Matthew Zook, Solon Barocas, Kate Crawford, Emily Keller, Seeta Peña Gangadharan, Alyssa Goodman, Rachelle Hollander, Barbara A Koenig, Jacob Metcalf, Arvind Narayanan, et al., "Ten simple rules for responsible big data research," *PLoS computational biology*, vol. 13, no. 3, pp. e1005399, 2017.

[117] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai, "Man is to computer programmer as woman is to homemaker? debiasing word embeddings," in *Advances in Neural Information Processing Systems*, 2016, pp. 4349–4357.

[118] Helen Nissenbaum, "How computer systems embody values," *Computer*, vol. 34, no. 3, pp. 120–119, 2001.

[119] Kenneth J Rothman, Sander Greenland, and Timothy L Lash, *Modern epidemiology*, Lippincott Williams & Wilkins, 2008.

[120] Yifan Hu, Yehuda Koren, and Chris Volinsky, "Collaborative filtering for implicit feedback datasets," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. Ieee, 2008, pp. 263–272.

[121] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims, "Recommendations as treatments: Debiasing learning and evaluation," *arXiv preprint arXiv:1602.05352*, 2016.

[122] Dawen Liang, Laurent Charlin, James McInerney, and David M Blei, "Modeling user exposure in recommendation," in *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016, pp. 951–961.

[123] Dawen Liang, Laurent Charlin, and David M Blei, "Causal inference for recommendation," .

[124] Mahsa Badami, Olfa Nasraoui, Wenlong Sun, and Patrick Shafto, "Detecting polarization in ratings: An automated pipeline and a preliminary quantification on several benchmark data sets," .

[125] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher, "Controlling popularity bias in learning-to-rank recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017, pp. 42–46.

[126] Allison JB Chaney, Brandon M Stewart, and Barbara E Engelhardt, "How algorithmic confounding in recommendation systems increases homogeneity and decreases utility," *arXiv preprint arXiv:1710.11214*, 2017.

[127] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin, "Unbiased offline recommender evaluation for missing-not-at-random implicit feedback," 2018.

[128] Ashudeep Singh and Thorsten Joachims, "Fairness of exposure in rankings," *arXiv preprint arXiv:1802.07281*, 2018.

[129] Ayan Sinha, David F Gleich, and Karthik Ramani, "Deconvolving feedback loops in recommender systems," in *Advances in Neural Information Processing Systems*, 2016, pp. 3243–3251.

[130] Patrick Shafto and Olfa Nasraoui, "Human-recommender systems: From benchmark data to benchmark cognitive models," in *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 2016, pp. 127–130.

[131] Thomas L Griffiths and Michael L Kalish, "Language evolution by iterated learning with bayesian agents," *Cognitive Science*, vol. 31, no. 3, pp. 441–480, 2007.

[132] Joshua Klayman and Young-Won Ha, "Confirmation, disconfirmation, and information in hypothesis testing.," *Psychological review*, vol. 94, no. 2, pp. 211, 1987.

[133] Vittorio Castelli and Thomas M Cover, "On the exponential value of labeled samples," *Pattern Recognition Letters*, vol. 16, no. 1, pp. 105–111, 1995.

[134] Richard S Sutton and Andrew G Barto, *Reinforcement learning: An introduction*, vol. 1, MIT press Cambridge, 1998.

[135] Jörg Rieskamp, Jerome R Busemeyer, and Barbara A Mellers, "Extending the bounds of rationality: evidence and theories of preferential choice," *Journal of Economic Literature*, vol. 44, no. 3, pp. 631–661, 2006.

[136] Donald B Rubin, "Inference and missing data," *Biometrika*, pp. 581–592, 1976.

[137] Patrick Shafto and Elizabeth Bonawitz, "Chapter four-choice from among intentionally selected options," *Psychology of Learning and Motivation*, vol. 63, pp. 115–139, 2015.

[138] Kelley Durkin, Leyla R Caglar, Elizabeth Baraff Bonawitz, and Patrick Shafto, "Explaining choice behavior: The intentional selection assumption.," in *CogSci*, 2015.

[139] Alan Stuart, J Keith Ord, and Sir Maurice Kendall, *Distribution theory*, Edward Arnold; New York, 1994.

[140] Pedro Domingos and Michael Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," *Machine learning*, vol. 29, no. 2-3, pp. 103–130, 1997.

[141] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant, *Applied logistic regression*, vol. 398, John Wiley & Sons, 2013.

[142] Corinna Cortes and Vladimir Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[143] F Maxwell Harper and Joseph A Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 5, no. 4, pp. 19, 2016.

[144] Michael E Tipping and Christopher M Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.

[145] Jacob Cohen, "Statistical power analysis for the behavioral sciences 2nd edn," 1988.

[146] Samuel Sanford Shapiro and Martin B Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, no. 3-4, pp. 591–611, 1965.

[147] Richard Lowry, "Concepts and applications of inferential statistics," 2014.

[148] Henry B Mann and Donald R Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The annals of mathematical statistics*, pp. 50–60, 1947.

[149] James Bennett, Stan Lanning, et al., "The netflix prize," in *Proceedings of KDD cup and workshop*. New York, NY, USA, 2007, vol. 2007, p. 35.

[150] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen, "Improving recommendation lists through topic diversification," in *Proceedings of the 14th international conference on World Wide Web*. ACM, 2005, pp. 22–32.

[151] Hadley Wickham, Doina Caragea, and Di Cook, "Exploring high-dimensional classification boundaries," in *Proceedings of the 38th Symposium on the Interface of Statistics, Computing Science, and Applications Interface 2006: Massive Data Sets and Streams*, 2006, pp. 24–27.

[152] Wenlong Sun, Olfa Nasraoui, and Patrick Shafto, "Iterated algorithmic bias in the interactive machine learning process of information filtering," in *KDIR*, 2018.

[153] Wenlong Sun, Sami Khenissi, Olfa Nasraoui, and Patrick Shafto., "Debiasing the human-recommender system feedback loop in collaborative filtering," in *Companion Proceedings of the 2019 World Wide Web Conference*. ACM, 2019.

[154] Guido W Imbens and Donald B Rubin, *Causal inference in statistics, social, and biomedical sciences*, Cambridge University Press, 2015.

[155] Scott Cheng-Hsin Yang, Jake Alden Whritner, Olfa Nasraoui, and Patrick Shafto, "Unifying recommendation and active learning for human-algorithm interactions," .

[156] Charles LA Clarke, Maheedhar Kolla, Gordon V Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon, "Novelty and diversity in information retrieval evaluation," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2008, pp. 659–666.

[157] David F Gleich and Lek-heng Lim, "Rank aggregation via nuclear norm minimization," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 60–68.

[158] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen, "Pearson correlation coefficient," in *Noise reduction in speech processing*, pp. 1–4. Springer, 2009.

[159] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 230–237.

[160] Benjamin Marlin, Richard S Zemel, Sam Roweis, and Malcolm Slaney, "Collaborative filtering and the missing at random assumption," *arXiv preprint arXiv:1206.5267*, 2012.

[161] Stephen J Wright, "Coordinate descent algorithms," *Mathematical Programming*, vol. 151, no. 1, pp. 3–34, 2015.

CURRICULUM VITAE

Wenlong Sun

Education

- Ph.D., University of louisville, Louisville,KY, May 2019
- Data Mining Certificate, University of Louisville, KY, Oct 2015
- M.S. Anhui university of Technology, Anhui China, Jun 2012
- B.S. Anhui university of Technology, Anhui China, Jun 2009

Work Experience

- Data science engineer III, Sojern Inc. (Feb 2019 - Present).
- Research Assistant, University of Louisville (Aug 2013 - May 2019).
- Computer Engineer & Data Scientist, Gen Nine Inc (May 2016 - Dec 2016).

Honors and Awards

- Best Paper Award at 10th international conference on Knowledge Discovery and Information Retrieval, Spain, Sep 2018
- Recipient of Dissertation Completion Award, Mar 2018
- Recipient of Prestigious Grosscurth Fellowship for Doctoral Student, Aug 2013
- Excellent Graduates of Anhui Province (top 5 percent), Anhui, China, Jun 2012

Publications and Presentations

- 1. Wenlong Sun, Olfa Nasraoui, Patrick Shafto., **Evolution and Impact of Bias in Human and Machine Learning Algorithm Interaction**. Under Revision. 2019.
- 2. Wenlong Sun, Olfa Nasraoui, Patrick Shafto., **User Polarization Aware Matrix Factorization for Recommendation System**. In preparation. 2019
- 3. Wenlong Sun, Sami Khenissi, Olfa Nasraoui, Patrick Shafto., **Debiasing the Human-Recommender System Feedback Loop in Collaborative Filtering**. Accepted in the international workshop on Augmenting Intelligence with Bias-Aware Humans-in-the-Loop, co-located with TheWebConf (WWW2019).
- 4. Wenlong Sun, Olfa Nasraoui, Patrick Shafto., **Iterated Algorithmic Bias in the Interactive Machine Learning Process of Information Filtering**. In Proceedings of the 10th International Conference on Knowledge Discovery and Information Retrieval - Volume 1: KDIR, pages 110-118, Spain 2018. (**Best Paper Award**, acceptance rate 18%)
- 5. Nutakki, Gopi Chand, Behnoush Abdollahi, Wenlong Sun, and Olfa Nasraoui. **An Introduction to Deep Clustering**. In Clustering Methods for Big Data Analytics, pp. 73-89. Springer, Cham, 2019.
- 6. Badami, Mahsa, Olfa Nasraoui, Welong Sun, and Patrick Shafto. **Detecting polarization in ratings: An automated pipeline and a preliminary quantification on several benchmark data sets**. 2017 IEEE International Conference on, pp. 2682-2690. IEEE, 2017..
- 7. Abdollahi, Behnoush, Mahsa Badami, Gopi Chand Nutakki, Wenlong Sun, and Olfa Nasraoui. **A two step ranking solution for twitter user engagement**. In Proceedings of the 2014 Recommender Systems Challenge, p. 35. ACM, 2014.
- 8. Nutakki, Gopi Chand, Olfa Nasraoui, Behnoush Abdollahi, Mahsa Badami, and

Wenlong Sun. **Distributed LDA-based Topic Modeling and Topic Agglomeration in a Latent Space**. In SNOW-DC@ WWW, pp. 17-24. 2014.

- 9. Wei, Xiaoli, Wenlong Sun, Xue Shi, Imhoi Koo, Bing Wang, Jun Zhang, Xinmin Yin et al. **MetSign: a computational platform for high-resolution mass spectrometry-based metabolomics**. Analytical chemistry 83, no. 20 (2011): 7668-7675.

- 10. Wang, Bing, Wenlong Sun, Jun Zhang, and Peng Chen. **Current status of machine learning-based methods for identifying protein-protein interaction sites**. Current Bioinformatics 8, no. 2 (2013): 177-182.

- 11. Zhong, Wei, Yantao Zhao, Yunan Tang, Xiaoli Wei, Xue Shi, Wenlong Sun, Xiuhua Sun et al. **Chronic alcohol exposure stimulates adipose tissue lipolysis in mice: role of reverse triglyceride transport in the pathogenesis of alcoholic steatosis**. The American journal of pathology 180, no. 3 (2012): 998-1007.

- 12. Wei, Xiaoli, Xue Shi, Wei Zhong, Yantao Zhao, Yunan Tang, Wenlong Sun, Xinmin Yin et al. **Chronic alcohol exposure disturbs lipid homeostasis at the adipose tissue-liver axis in mice: analysis of triacylglycerols using high-resolution mass spectrometry in combination with in vivo metabolite deuterium labeling**. PloS one 8, no. 2 (2013): e55382.

- 13. Zhong, Wei, Yantao Zhao, Yunan Tang, Xiaoli Wei, Xue Shi, Wenlong Sun, Xiuhua Sun et al. **chronic Alcohol Exposure Causes Adipose Fat Overflux To The Liver In Mice: A Mechanistic Link Between Lipodystrophy And Steatosis: 1313**. Hepatology 54, no. 4 (2011): 978A.