Towards user-oriented privacy for recommender system data
A personalization-based approach to gender obfuscation for user profiles

Slokom, Manel; Hanjalic, Alan; Larson, Martha

DOI
10.1016/j.ipm.2021.102722

Publication date
2021

Document Version
Final published version

Published in
Information Processing and Management

Information Processing and Management 58 (2021) 102722

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/ipm

Towards user-oriented privacy for recommender system data: A personalization-based approach to gender obfuscation for user profiles

Manel Slokom [a,*], Alan Hanjalic [a], Martha Larson [b,a]

[a] Delft University of Technology, Netherlands
[b] Radboud University, Netherlands

ARTICLE INFO

Keywords:
Top-N recommendation
Obfuscation
Gender inference
Evaluation
Privacy
Fairness
Diversity

ABSTRACT

In this paper, we propose a new privacy solution for the data used to train a recommender system, i.e., the user–item matrix. The user–item matrix contains implicit information, which can be inferred using a classifier, leading to potential privacy violations. Our solution, called Personalized Blurring (PerBlur), is a simple, yet effective, approach to adding and removing items from users' profiles in order to generate an obfuscated user–item matrix. The novelty of PerBlur is personalization of the choice of items used for obfuscation to the individual user profiles. PerBlur is formulated within a user-oriented paradigm of recommender system data privacy that aims at making privacy solutions understandable, unobtrusive, and useful for the user. When obfuscated data is used for training, a recommender system algorithm is able to reach performance comparable to what is attained when it is trained on the original, unobfuscated data. At the same time, a classifier can no longer reliably use the obfuscated data to predict the gender of users, indicating that implicit gender information has been removed. In addition to introducing PerBlur, we make several key contributions. First, we propose an evaluation protocol that creates a fair environment to compare between different obfuscation conditions. Second, we carry out experiments that show that gender obfuscation impacts the fairness and diversity of recommender system results. In sum, our work establishes that a simple, transparent approach to gender obfuscation can protect user privacy while at the same time improving recommendation results for users by maintaining fairness and enhancing diversity.

1. Introduction

The data used to train a recommender system takes the form of a user–item matrix, where the columns represent items in the collection and the rows represent individual users. Each row contains a user's item ratings, or item interactions, and is referred to as a user profile. The user–item matrix does not explicitly contain specific user attributes such as gender. However, such information is implicit in each profile, since it can be predicted or inferred using machine learning, specifically, a classifier. This information represents a privacy threat for users.

As the user profiles collected and stored by online platforms increase in number and length, classifiers have a larger amount of data available for training and inference, and the privacy threat grows. To counter this threat, we need the right privacy solutions. Less obviously, we need to re-examine our underlying assumptions about user privacy and to be open to a variety of paradigms.

* Corresponding author.
E-mail addresses: m.slokom@tudelft.nl (M. Slokom), a.hanjalic@tudelft.nl (A. Hanjalic), m.larson@cs.ru.nl (M. Larson).

https://doi.org/10.1016/j.ipm.2021.102722
Received 30 November 2020; Received in revised form 30 July 2021; Accepted 4 August 2021
Available online 14 September 2021

Table 1
User-oriented paradigm for privacy of recommender system data. This paradigm forms the basis for the design of our approach.

Desideratum	Description
Understandable	User understands why items have been added to or removed from the profile.
Unobtrusive	Obfuscation should not be pure "noise", but rather be consistent with the user's own preferences.
Useful	Maintain or enhance recommendation performance, fairness, diversity

In this paper, we propose a privacy protection solution for the user–item matrix called Personalized Blurring (PerBlur),[1] which applies individualized obfuscation to user profiles. Obfuscation is a privacy protection approach that uses small changes to mask sensitive information. Our solution is formulated within a user-oriented paradigm of recommender system data privacy, which strives towards privacy that is *understandable*, *unobtrusive*, and *useful* for the user. In Section 1.1, we explain the paradigm, present a comparison and contrast with previous work, and motivate our PerBlur approach. In Section 1.2, we present our threat model, which formalizes the types of scenarios to which PerBlur applies. Specifically, PerBlur addresses privacy for cases in which an attacker is able to gain control of the entire data set, as occurs with a data breach or with drifting of goals, known as "mission creep". In Section 1.3, we explain the experimental framework. Our paper presents extensive experimental analysis of PerBlur, focusing on its usefulness for the user in terms of recommender system performance, fairness, and diversity. In this work, we focus on gender obfuscation, but PerBlur would also be suited for protecting other sorts of information that can be inferred from user profiles.

The paper makes the following contributions.

- We introduce PerBlur (Section 3) and demonstrate its ability to effectively obfuscate user profiles to protect information on user gender (Section 5).
- We propose an evaluation process for obfuscated recommender system data that addresses the challenges of comparing the performance of recommender systems trained on data that has been obfuscated in different ways (Section 6.1).
- We show that training recommender systems on obfuscated data leads to little, if any, loss in the quality of the recommendations received by the user, i.e., recommendation performance and that PerBlur is particular effective at maintaining recommender system performance (Section 6.2).
- We show the interplay between user-profile obfuscation and fairness (Section 7) and diversity (Section 8) and demonstrate the potential of PerBlur to contribute in both cases.

Taken together, our experimental analysis constitutes compelling evidence that user-oriented privacy can be achieved with an obfuscation-based method that is useful to users, while remaining understandable and unobtrusive. To our knowledge, our work represents the most convincing case to date for recommender system data privacy within a strongly user-oriented paradigm that prefers simplicity and transparency over formality and complexity.

1.1. User-oriented paradigm for privacy protection

The user-oriented paradigm for privacy protection expresses the requirements and priorities underlying our approach to addressing privacy threats that arise when users share their interaction data and recommender system platforms store these data as user profiles. The idea at the foundation of our paradigm is that privacy protection should center on users, serving their needs and allowing them to maintain insight and control. The idea of user-oriented privacy, defined in this way, has been around for at least a decade already in a somewhat weaker form. Two key examples of user-oriented approaches to protecting user profiles are Berkovsky, Kuflik, and Ricci (2012), which studies the impact of obfuscation without attempting to protect a specific user attribute, and Weinsberg, Bhagat, Ioannidis, and Taft (2012) (BlurMe), which is designed to protect the specific attribute of gender. These contributions make the assumption that users should have some measure of control over the obfuscation of their own profiles.

In this work, we move the design of user-oriented privacy beyond the orientation towards user control, to include other desirable, user-oriented characteristics. Specifically, the user should find privacy protection to be *understandable*, *unobtrusive*, and *useful*. The characteristics are the basis of the design of our personalization-based approach to gender obfuscation for recommender system data.

Our paradigm is summarized in Table 1, and next we will discuss each desirable characteristic in turn. The discussion will shed light on the advantage offered by privacy approaches that prefer simplicity and transparency over formality and complexity.

[1] GitHub Link: https://github.com/SlokomManel/PerBlur.

1.1.1. Obfuscation should be understandable

The *understandable* dimension of our paradigm expresses the importance that our paradigm places on approaches that the user can understand. The dimension is based on basic observations about how people protect their own privacy in offline environments. When we are offline, we protect our own privacy by choosing what we reveal about ourselves and whom to reveal it to. Our choices are based on our intuition and experience of what we can share without getting hurt, and we are not concerned with formal guarantees.

Our paradigm aims to maintain this natural approach to privacy in the online world. We strive for privacy protection that is conceptually simple so that people can form intuitions about it, allowing them to understand, or even choose, information that has been added to or subtracted from their profiles in order to achieve obfuscation. Our approach, PerBlur, is based on the idea, originating from BlurMe of Weinsberg et al. (2012), that to obfuscate gender, we should simply extend a user's profile with items that are indicative of the opposite gender. For example, in the movie domain, "Gone with the Wind" is indicative of female users and "Apocalypse Now" is indicative of male users. It is completely transparent to a male user how adding "Gone with the Wind" to his profile will obfuscate his gender.

Our work stands in contrast to paradigms which emphasize formal guarantees. An example is Yang, Qu, and Cudré-Mauroux (2019), which minimizes privacy leakage under a bound of the negative impact on the recommender system ranking. In this work, minimizing privacy leakage is achieved at the cost of the assumption of the existence of a detailed user profile specifying the information to be leaked. In contrast, PerBlur applies to any user profile without detailed knowledge of the user.

Our experiments demonstrate that it is possible to achieve successful obfuscation and simultaneously maintain recommender system performance with a "rough and ready" choice of an operating point, i.e., by estimating the necessary amount of obfuscation at the level of the collection rather than via a process of iterative optimization. The success of this "rough and ready" approach is quite remarkable, since the current trend is to immediately assume that obfuscation challenges require iterative optimization, i.e., using Generative Adversarial Networks (GANs). In Beigi, Mosallanezhad, Guo, Alvari, Nou, and Liu (2020), a GAN-based approach to protecting user attributes while maintaining recommender performance is proposed. The work is not directly comparable to our own, since the authors address a different threat model. Our own threat model, which is more formally specified in Section 1.2, protects information in the user item matrix. In contrast, Beigi et al. (2020) protects a combination of the user embeddings and the recommender output. However, this paper is relevant because it shows that we cannot assume that data obfuscated using a GAN-based approach will be capable of enabling the level of recommendation performance achieved using original data. Specifically, the GAN in Beigi et al. (2020) does not quite reach the precision and recall of the system before obfuscation. With our experiments, we will show that PerBlur, using its "rough and ready" approach to hyperparameter setting, gets very close to the performance with the original data, and in some cases surpasses it. At the same time, PerBlur obfuscation is understandable to the user and it also does not have to be recomputed from scratch as the user continues to rate or interact with items and the profile grows.

We also note that Beigi et al. (2020) claims that their approach outperforms BlurMe (Weinsberg et al., 2012). However, the support for the claim is weak. In Weinsberg et al. (2012), it is shown that BlurMe can achieve the recommendation performance of achieved using the original, unobfuscated data. We also reach this conclusion on the basis of our experiments. In contrast, (Beigi et al., 2020) lacks discussion of why their implementation of BlurMe falls very far short of the original data in ability to maintain recommendation performance. A possible explanation is that Beigi et al. (2020) does not adapt BlurMe for their threat model, which would be necessary in order to achieve a fair comparison.

1.1.2. Obfuscation should be unobtrusive

The *unobtrusive* characteristic expresses the commitment of our paradigm to approaches that do not hamper or otherwise inconvenience or disturb the user. In other words, the user should not perceive the protection as getting in the way. This requirement is in line with previous work Berkovsky et al. (2012), Chen, Boreli, Kaafar, and Friedman (2014) that has carried out user evaluation to test whether recommendations using the obfuscated matrix affect the satisfaction of the users.

Here, we incorporate our concern with unobtrusiveness into the design of the obfuscation. Specifically, we strive to make obfuscated profiles remain as natural as possible. PerBlur goes beyond BlurMe (Weinsberg et al., 2012) with respect to the goal of naturalness. Specifically, PerBlur does not draw heavily on the most indicative movies of the opposite gender. For example, "Gone with the Wind" could be used to obfuscate some user profiles, but if every male looking to hide his gender had "Gone with the Wind" in his profile, the obfuscation would become obvious. PerBlur also avoids the larger issue is that a male user might not want to have a particular movie in his profile. For example, "Gone with the Wind" romanticizes the US Civil War, and, today, its depictions of the South are understood as racist. A user obfuscating his profile would prefer to have movies that are consistent with his tastes.

PerBlur achieves unobtrusiveness by *personalizing* obfuscation so that it matches the preferences of the user being obfuscated as well as possible. Specifically, the items that extend a user's profile are both indicative of the opposite gender and, at the same time, reflective of the user's preferences. Our paradigm stands in contrast with the paradigm used by nearly every other research effort in the direction of obfuscation for privacy in recommender systems, which obfuscate by introducing noise into the user data. For example, Differential Privacy is explicitly directed at adding noise to user profiles. An example of such an approach is Friedman, Berkovsky, and Kaafar (2016). We do not consider such approaches user-oriented since they miss the chance to attempt to align obfuscation with user preferences.

Table 2
Threat model: Gender inference on user–item data used for recommender systems.

Component	Description
Adversary: Resources	The attacker has a gender classifier pre-trained on unobfuscated data or has the data necessary to train one.
Adversary: Objective	The inference of users' gender attribute.
Vulnerability: Opportunity	The possession of a user–item matrix.
Vulnerability: Countermeasure	Obfuscation of the user–item matrix to block the inference of gender.

1.1.3. Obfuscation should be useful

The *useful* dimension expresses the commitment of our paradigm to serving users needs. First, obfuscation should strive to maintain recommender performance, i.e., the accuracy of the recommended items from the perspective of the user. Most other work on recommender system privacy, agrees on this point. However, within our paradigm we go beyond accuracy. We are also interested on maintaining the usefulness of the recommendations with respect to fairness and diversity. To our knowledge, we are the first work to experimentally demonstrate that recommender data obfuscation can impact the fairness and diversity of recommender systems trained on that data.

1.2. Threat model for gender inference

Our goal is to protect user privacy in the case that recommender system data, i.e., the entire user–item matrix, falls into the hands of a party whose goal is to infer gender information about individual users. We call this party the *attacker*. In this section, we specify our goal more formally in the form of a threat model.

We start with some general comments about the conditions under which an attack might occur. Perhaps the most obvious way in which the attacker can acquire the entire user–item matrix is via a breach of the recommender platform. However, it is also possible that the attacker is internal to the platform. For example, a platform might collect user data without the intention to infer gender information. However, the business strategy of the company owning the platform might change, or the company might be bought by another company. In this case, so-called "mission creep" can occur. In other words, the data is used for something other than the original purpose. It is important to note that the privacy threat that we are addressing differs from that addressed by the large portion of the literature on recommender system privacy, summarized for example by Friedman, Knijnenburg, Vanhecke, Martens, and Berkovsky (2015). Work such as Berkovsky et al. (2012), Parameswara and Blough (2007), Polat and Du (2003) often aims to improve the privacy of users, but under the assumption that the platform does not lose control of user data. Work such as Anelli, Deldjoo, Di Noia, Ferrara, and Narducci (2021), Badsha, Yi, Khalil, and Bertino (2017), Qiang (2019) adopts a federated learning approach, which assumes the existence of clients, which can also be breached individually.

Our threat model serves to make the scenario we address concrete, and clearly differentiate it from scenarios addressed by other work. Such a threat model is generally used in security and privacy research, and specifies the conditions for which protection is developed and against which protection is tested. Our model is presented in Table 2.

The threat model follows the main dimensions set out in Salter, Saydjari, Schneier, and Wallner (1998). First, it describes the adversary, including the resources at the adversary's disposal and the adversary's objective. In other words, the threat model specifies what the attacker is capable of and what the goal of the attacker is. Second, it describes the vulnerability, including the opportunity that makes an attack possible, and the nature of the countermeasures that can be taken to prevent the attack.

Table 2 provides the specifications of our threat model for each of the dimensions. As resources, we assume that the attacker has a gender classifier that is pre-trained on unobfuscated data. The objective is to infer the gender of individual users. The data is unobfuscated because we assume that the attack is *blackbox* in the sense that the attacker does not have access to information about the obfuscation. In our experiments, the gender inference classifier is trained using data drawn from the same sources as the user profiles that are subject to attack. This means that our attack is somewhat stronger than what could be expected in the real world, where the attacker would not necessarily have access to data from the same source.

The opportunity for attack is the possession of the entire user–item matrix. We note that anonymization is important but here we are not interested in whether attackers can reconstruct the identity of the users, but rather whether they can infer a gender for each user-ID. Finally, the countermeasure that we are investigating is obfuscation. Note that our focus on obfuscation does not imply that other countermeasures may not be important. For example, encryption protects privacy in the case of a data breach. However, we focus on obfuscation because users' data might actually be partially public, for example, on a social media website, and because encryption does not address the issue of mission creep.

We finish this section with some additional discussion on why we do not strive for privacy with formal guarantees. As previously stated, privacy in the real-world does not offer guarantees. Further, our experiments will show that the trade-off in privacy vs. protection is small, if it exists at all. The implication is that the user can have intuitive confidence without needing a guarantee, circumventing the question of whether the guarantee is understandable. Another interesting consideration is that formal guarantees

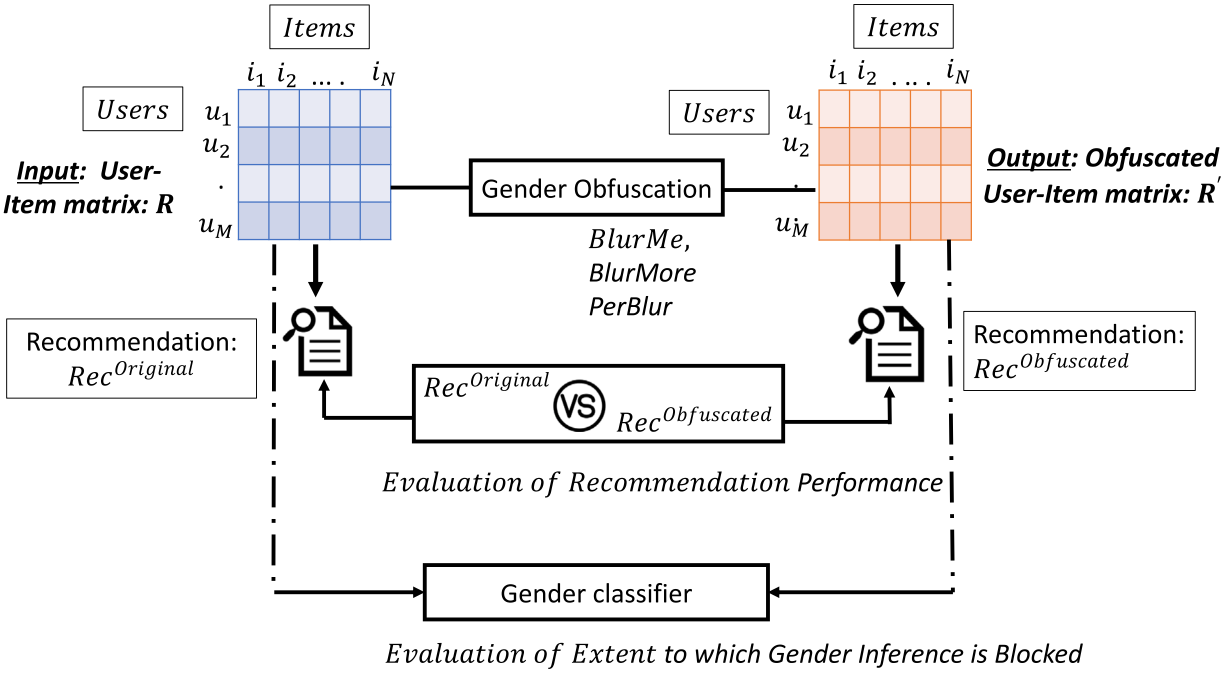

Fig. 1. Gender obfuscation of recommender system data (user–item matrix). Evaluation involves comparing recommendation performance on original and obfuscated data and also confirming the extent to which the inference of user gender from the obfuscated data is reduced or prevented.

cover *defenses* but not *meta-defenses*. In other words, formal guarantees capture the degree to which attacks are blocked, but do not cover the goal of motivating the attacker to give up entirely. In a practical situation, we should be interested not only in ensuring that attackers be unsuccessful in inferring gender, but in nudging them to abandon the effort of inferring gender. For example, the incentive for mission creep within a recommender system platform towards gender inference evaporates if gender inference requires large amounts of resources and yields only low quality information. We do not consider meta-defense further here, but mention the issue only for completeness.

1.3. Experimental framework

Next we present the framework that we use to carry out our analysis of gender obfuscation and demonstrate the properties of our PerBlur approach. As shown in Fig. 1 (top), gender obfuscation takes the original user–item matrix $\mathcal{R}$ and transforms it into the obfuscated user–item matrix $\mathcal{R}'$. In order to be successful, gender obfuscation must fulfill two criteria. First, as indicated by "Evaluation of Recommendation Performance" in Fig. 1 (middle), the quality of the predictions produced by the recommender system must be comparable for the original and the obfuscated data. Second, as indicated by "Evaluation of the Extent to Which Gender Information is Blocked" in Fig. 1 (bottom), a gender classifier must no longer be able to use the obfuscated data to reliably predict the genders of the users.

In addition to studying recommendation performance, our experiments also analyze obfuscated data with respect to its ability to support the fairness and diversity of recommendations.

2. Background and related work

In this section, we first give a brief overview of existing work on privacy in recommender systems. Then, we cover previous work on obfuscation for privacy. Next, we provide background on gender inference, and, finally, we discuss related work on fairness and diversity.

2.1. Privacy in recommender systems

Privacy-preserving techniques for recommender systems can be understood as falling into different groups (Jeckmans, Beye, Erkin, Hartel, Lagendijk, & Tang, 2013). Here, we discuss several key examples of those groups. *Indistinguishability-based techniques* (Casino, Domingo-Ferrer, Patsakis, Puig, & Solanas, 2015), such as k-anonymity, l-diversity, and t-closeness, are designed to protect against re-identification attacks. *Differential-based techniques* aim to obscure the link between a user's information in the input (the user's preferences) and output (the recommendation) (Dwork, 2008; Jeckmans et al., 2013). McSherry and Mironov (2009), Hua, Xia, and Zhong (2015) and Friedman et al. (2016) proposed different ways to apply differential privacy to matrix factorization that can prevent an untrusted recommender from learning specific user preferences. Hua et al. (2015) added noise to item vectors to make them differentially private. Friedman et al. (2016) perturbed the input data by introducing noise prior to the data analysis. *Data masking techniques* (Parra-Arnau, Rebollo-Monedero, & Forné, 2014; Polat & Du, 2003) obfuscate users' information by perturbing the input data. Kandappu, Friedman, Boreli, and Sivaraman (2014) proposed "Privacy Canary", an interactive system that enables users to interact and control the privacy-utility trade-off of the recommender system to achieve a desired accuracy while maintaining privacy protection.

In this work, we are not interested in general privacy, but rather in protecting recommender system data. Different techniques have been proposed to protect recommender system data. Some techniques approach the problem from a security point of view. Focusing on securing the *system*, Burke, O'Mahony, and Hurley (2015), Deldjoo, Di Noia, and Merra (2021), Mobasher, Burke, Bhaumik, and Williams (2007) attempt to prevent attackers from manipulating the recommendation results through the insertion of fake user profiles called profile injection attack. The objective of a profile injection attack is to promote (called item push) or demote (called item nuke) the recommendations made for specific items (Burke et al., 2015). Badsha, Yi, and Khalil (2016) and Nikolaenko, Ioannidis, Weinsberg, Joye, Taft, and Boneh (2013) proposed to protect matrix factorization by applying homomorphic encryption that provides recommendations without knowing the actual ratings. Other techniques focus on protecting recommender system data in order to improve privacy, i.e., prevent the disclosure of users' information. It is important to differentiate between work that protects information implicit in the user–item matrix, such as Slokom, Larson, and Hanjalic (2019), from work that protects the information implicit in the list of recommendations (Beigi et al., 2020; Calandrino, Kilzer, Narayanan, Felten, & Shmatikov, 2011). Different disclosure attacks have been studied. Here we differentiate between re-identification attacks (Narayanan & Shmatikov, 2008) and inference attacks (Gong & Liu, 2018; Kosinski, Stillwell, & Graepel, 2013). In our work, we are interested in protecting the user–item matrix against inference attacks.

2.2. Obfuscation for privacy

2.2.1. Data obfuscation

Data obfuscation is a privacy preserving technique that aims to hide sensitive information in the data by adding ambiguous, confusing, or misleading information (Brunton & Nissenbaum, 2015) in order to prevent inference attacks and sensitive information leakage. Obfuscation can be applied to different domains with different input data such as online social networks (Chen et al., 2014), location-based services (Ardagna, Cremonini, Damiani, Di Vimercati, & Samarati, 2007), photos (Li, Vishwamitra, Knijnenburg, Hu, & Caine, 2017), text (Reddy & Knight, 2016), and recommender systems (Berkovsky et al., 2012; Feng, Guo, & Chen, 2015b; Parameswara & Blough, 2007; Strucks, Slokom, & Larson, 2019; Weinsberg et al., 2012).

In this paper, we focus on data obfuscation for recommender systems research. Our work is most closely related to the following papers. Berkovsky et al. (2012) focused on enhancing the privacy of recommender system users by distributing their profiles across multiple repositories and then, obfuscating the user profiles to partially hide the actual user ratings. Berkovsky et al. investigated three data obfuscation strategies: (1) Default obfuscation replaces real ratings in the user profile with a predefined value, (2) uniform random obfuscation replaces real ratings with random values chosen within the range of ratings, (3) distribution-based obfuscation replaces real ratings with values drawn from the distribution of ratings in the data set. In Parameswara and Blough (2007) a privacy preserving framework is proposed to make it possible for multiple E-commerce services to share data. The data sets are obfuscated by permuting sets of similar items.

We note that obfuscation is different from injection attacks. Obfuscation and injection (shilling) attacks are similar in the sense that they both manipulate the user profile but with different goals. Obfuscation focuses on protecting users' information existing in the user–item matrix (a defense technique) but injection attacks are generally techniques for attacking the recommender systems.

2.2.2. Gender obfuscation

Gender Obfuscation is a subset of data obfuscation, which aims to protect the privacy of users, while maintaining the utility of the data. Specifically, obfuscation has the goal of making it more difficult to infer the gender of the user from data using a classifier. Gender obfuscation is widely studied as a surrogate for obfuscating other sensitive information such as age or profession.

Gender obfuscation for recommender system data was originally proposed by Weinsberg et al. (2012). This work showed that we can infer the binary gender of a user with high accuracy, based solely on recommender system data (i.e., rated movies). They then proposed an algorithm, called *BlurMe*, which obfuscates the user–item matrix in a way that blocks this inference while maintaining the performance of rating prediction. The basic idea is to add fictional item ratings to every user profile that are typical for the opposite gender. Tests of BlurMe involved only obfuscating 10% of the data at a time, so the goal was not directly to protect the entire user–item matrix as we attempt to do here.

After BlurMe, Feng et al. (2015b) introduced a privacy preserving module (called PP module) situated between the recommender system and the user. PP module also adds a set of extra fictitious ratings of items not rated by the given user. Although Feng et al. (2015b) moves away from the one-size-fits all obfuscation used by BlurMe, it does not propose to leverage imputation for personalized obfuscation, as we do in this paper. Further, Feng et al. (2015b) focused on rating prediction and did not propose approaches for Top-N prediction, as we do here. In Yang et al. (2019), an approach to obfuscating an entire user–item matrix was proposed, however, this approach necessitates the use of detailed private data from users to determine whether privacy is being leaked.

In previous work, notably BlurMe (Weinsberg et al., 2012), the goal has been to reduce the accuracy as far as possible. This goal is not particularly helpful to privacy protection. If the accuracy of a binary gender classifier is very low, and if the attacker realizes that the data has been protected, then it is possible to recover reliable gender predictions by simply flipping the classifier decision. In our work, we adopt the position that once the AUC performance has been reduced to 0.5 (where there is not benefit from a flip), then, we have succeeded to block gender classification and it is not necessary to reduce it lower.

There have been a number of approaches to gender obfuscation related to recommender systems. It is important to note, however, that these approaches differ from our work because they are protecting an aspect of the recommender system other than the data, as we do here. We mention these approaches here for completeness. Resheff, Elazar, Shahar, and Shalom (2018) showed that private demographic information can be leaked via the user representations used by latent factor recommender systems. Resheff et al. adopted an adversarial training framework with which they simultaneously perturb the user vectors in order to harm the readout of the private information and change the recommender parameters until the system is optimized. As mentioned above, Hu and Yang (2020) adopted an adversarial learning technique to learn a privacy-aware transfer model. The generator represents the attacker who tries to infer the user privacy, while the discriminator is the recommender which learns user preferences and deceives the adversary. In this work, Hu et al. focus on perturbing the representations of the system, rather than the recommender system data, as we do here. Note that in our work the obfuscation approach and the classifier can be considered to stand in an adversarial relationship. However, we do not optimize them together, as would be done with a GAN.

Note that there is some work on gender obfuscation outside of recommender systems. In particular, we mention Chen et al. (2014), which focused on online social networks. Chen et al. (2014) studied how the adoption of different obfuscation strategies e.g., addition, removal or replacement by different proportions of users affects the inference attacks. We mention this work to demonstrate the viability of obfuscation approaches to privacy.

2.3. Gender inference

We use the term inference attack to refer to the use of an inference algorithm to infer something that a user may consider private i.e., age, gender, or sexual orientation. Most of the users are not aware of the correlation that exists between their public and private data (Salamatian, Zhang, du Pin Calmon, Bhamidipati, Fawaz, Kveton, Oliveira, & Taft, 2015). For example, just from Facebook Likes (Kosinski et al., 2013) or ratings given to consumed items (Chen et al., 2014; Feng, Guo, & Chen, 2015a; Salamatian et al., 2013; Weinsberg et al., 2012), an attacker can accurately predict a range of highly sensitive personal attributes including (Salamatian et al., 2015): sexual orientation, ethnicity, religious and political views, age, and gender.

Some authors (Jia, Wang, Zhang, & Gong, 2017; Mislove, Viswanath, Gummadi, & Druschel, 2010) have studied the problem of inference of user attributes in online social networks. Jia et al. (2017) proposed a method called "AttriInfer" that combines both friends and behaviors in a social graph. AttriInfer illustrated that even when only a fraction of users provide publicly their profile attributes (such as location, interests), it is possible to infer these attributes among users who do not disclose them. Bi, Shokouhi, Kosinski, and Graepel (2013) showed how user demographic traits such as age, gender, and even political and religious views can be inferred based on their search query histories. Bhagat, Weinsberg, Ioannidis, and Taft (2014) presented a new inference attack that a recommender system could use to infer demographic attributes for private user profiles. In the area of online video systems, a gender inference algorithm (Feng et al., 2014) was used to infer viewers' gender based on implicit watch history.

2.4. Fairness and diversity

The goal of fairness is to design algorithms that make fair predictions across various (i.e., demographic) groups (Friedler, Scheidegger, & Venkatasubramanian, 2021; Yao & Huang, 2017). There are different kinds of fairness (Abdollahpouri et al., 2019; Burke, Sonboli, & Ordonez-Gauger, 2018; Ekstrand, Joshaghani et al., 2018): *consumers fairness* (C-fairness): where the recommendations should be fair towards the users in the protected class (as defined by gender, age, nationality, ethnicity, etc.) relative to other users. *Providers fairness* (P-fairness) treat the providers of the items in a fair way (Ferraro, Serra, & Bauer, 2021), and *multi-sided fairness* (CP-fairness) (Abdollahpouri et al., 2019; Burke, 2017) requires fairness to be considered for both consumers and providers. Ekstrand et al. (2018) looked at C-fairness by exploring whether different user demographic groups experience similar or different utility from the recommendation system. Ekstrand et al. proposed an empirical analysis of the effectiveness of collaborative filtering recommendation strategies stratified by the gender and age of the users. They found that not all users experience the system in the same way. Mansoury, Abdollahpouri, Smith, Dehpanah, Pechenizkiy, and Mobasher (2020b), explored different factors (e.g., the user profile size, the entropy of users profiles and the anomaly in rating behavior) that could be associated with the unfairness of performance of recommendation algorithms for males versus females. They showed that neighborhood-based algorithms such as UserKNN and ItemKNN discriminate more against female users. To address provider fairness, Ekstrand and Kluver (2021), Ekstrand, Tian, Kazi, Mehrpouyan, and Kluver (2018) looked at the response of collaborative filtering recommender algorithms to the distribution of their input data with respect to the content creator gender. In the context of book recommendation, Ekstrand et al. investigated how recommender systems interact with author gender in book data. In the context of music recommendation, Shakespeare, Porcaro, Gómez, and Castillo (2020) studied the extent to which collaborative filtering recommendation algorithms may increase or decrease artist gender bias. Epps-Darling, Bouyer, and Cramer (2020) studied gender representation in music streaming. They found that listeners generally tend to stream fewer female artists than male artists.

Here, we focus on consumer fairness (C-Fairness). Specifically, we are worried about the recommendation system performing well for users of one gender and not for another. We followed the same measures used in Ekstrand et al. (2018).

Diversity in recommender systems has been broadly studied in literature (Castells, Hurley, & Vargas, 2015; Hansen, Mehrotra, Hansen, Brost, Maystre, & Lalmas, 2021; Kaminskas & Bridge, 2016; Kunaver & Požrl, 2017; Vargas & Castells, 2011). Generally, diversity applies to a set of items and it has to do with how different the items are with respect to each other (Castells et al., 2015). Hansen et al. (2021) aimed at shifting users' consumption towards the tail and less familiar content in the context of music streaming. Hansen et al. defined diversity around two factors that influence the consumption of music. First, the *taste similarity* which means how similar a piece of music is to the type of music the user has listened to previously. Second, *popularity* or how many users have recently listened to the piece of music. Mansoury, Abdollahpouri, Pechenizkiy, Mobasher, and Burke (2020a) proposed a graph-based approach, FairMatch, that works as a post-processing approach after recommendation generation for improving the aggregate diversity. Aggregate diversity is defined in literature as long-tail recommendation, which refers to the fact that the recommender systems should recommend a wide variety of items across all users. FairMatch improved the visibility of high-quality items that have a low visibility in the original set of recommendations. Oliveira, Nóbrega, Marinho, and Andrade (2017) proposed a multiobjective optimization solution for music recommendations that are at the same time diverse and similar to user preferences. The recommended lists aim at balancing between the aspects that should be held fixed (maximize similarity with users actual items) and aspects that should be diversified (minimize similarity with other items in the recommendation list). Vargas and Castells (2011) defined novelty and diversity based on three key concepts namely choice, discovery and relevance. Helberger, Karppinen, and D'Acunto (2018) highlighted a number of principles designed for exposure diversity in recommender systems.

Here we study diversity with respect to gender specificity. We look at gender specificity and investigate how to control the number of gender-stereotypical items recommended to users. Our goal is preventing users from getting overrun with items that are stereotypical for their gender. For example, a woman might want to watch one Hallmark Christmas romance movie, and if a recommender system diversifies for gender specificity, it will prevent her recommendation list from being flooded with other Hallmark Christmas romances. In this paper, the study of gender-stereotypical items diversity is different from popularity. In gender-stereotypical items we compare the recommended items vs. items highly indicative for female (or male) users. There is no direct relation between the list of indicative items and the popularity of items.

2.5. Imputation for user–item matrices

Imputation approaches are approaches used to fill in the missing values of user–item matrices (Bertsimas, Pawlowski, & Zhuo, 2017; Lakshminarayan, Harp, & Samad, 1999). The goal of the approaches is to infer missing values in data set in such a way that improves the overall performance of recommender systems trained on that data set (Su, Khoshgoftaar, & Greiner, 2008). Su et al. (2008) proposed two neighborhood based collaborative filtering imputation algorithms called imputed nearest neighborhood CF (INN-CF) and imputed densest neighborhood CF (IDN-CF). INN-CF first finds the most similar users to the target user. Then, it uses the corresponding imputed rating data to make predictions. IDN-CF makes predictions from the imputed densest neighbors (i.e., the users who have rated the most number of items).

In our work, we use imputation to personalize obfuscation. Specifically, we impute in order to derive a confidence score that allows us to choose the items that are added to the profile and also to (in the case of rating data) predict the rating that those items should have. Our choice of imputation is inspired by Su et al. (2008). We point out that our main goal is to obfuscate data, but that imputation actually has the goal of increasing recommender performance. For this reason, we can expect that PerBlur might actually be able to increase recommendation performance.

Evidence of the benefits of imputation has been given by Su et al. (2008), who found that imputation boosts the predictive performance for collaborative filtering recommendations. Another example of a related paper that used imputation to improve performance is Yuan, Han, Qian, Xu, and Yan (2019), which proposed a novel method ISVD to incorporate imputed data into SVD framework. For imputation, ISVD chooses effective neighbors for the users and items based on the similarity relation among users and items. The imputed ratings are produced and then incorporated into the SVD model. Imputation can also provide benefit when used for augmentation. The work in Li, Zheng, et al. (2017), Wu, DuBois, Zheng, and Ester (2016) introduced a sparsity-aware data-augmentation strategy that provides more item correlation patterns and hence improves recommendation performance.

3. Personalized blurring (PerBlur)

In this section, we present a basic skeleton for gender obfuscation and also introduce *PerBlur*, our approach to gender obfuscation for recommender system data. The main idea of PerBlur is to obfuscate the gender of a user in the user–item matrix by extending the user's profile in a personalized manner, while simultaneously ensuring that the extension is not typical for the user's gender. Specifically, the standard PerBlur algorithm adds ratings (or interactions) to a user's profile that are consistent with the user's preferences, but are at the same time indicative for the opposite gender. PerBlur has two variants: The standard variant just adds ratings (or interactions), and the variant "PerBlur with removal" removes ratings (or interactions) that are indicative for the user's own gender.

Recall that PerBlur builds on the basic idea of BlurMe (Weinsberg et al., 2012), which is to obfuscate by adding indicative items for the opposite gender. In our work, BlurMe is also applied differently from the original BlurMe paper (Weinsberg et al., 2012). First, we are focused on studying Top-N recommendation, whereas Weinsberg et al. (2012) studies exclusively rating prediction. Second, our goal is to protect the entire data set, and we apply obfuscation to all user profiles. In contrast, the goal of Weinsberg et al. (2012) is to protect individual users and in Weinsberg et al. (2012) obfuscation is applied only to 10% of the data at a time.

In our experiments, we show, for the first time, that the basic BlurMe can maintain recommender system performance in the case of Top-N recommendation and also in the case that the entire data set is obfuscated.

PerBlur also builds on the idea of our own previous (preliminary) work, BlurM(or)e (Strucks et al., 2019), which removes ratings to make the additional ratings less obvious and to prevent the user–item matrix from becoming dense, resulting in more naturalistic data. PerBlur introduces innovations beyond BlurMe and BlurM(or)e in two respects: It personalizes the extension of the user profile (personalization) and it also prioritizes the items to remove so that the most typical items for a user's gender are removed first (greedy removal).

Before presenting the details of PerBlur, we first present the basic skeleton of the gender obfuscation, which we will use in our experiments for BlurMe and PerBlur in order to compare the two approaches. Input to the algorithm is the level of obfuscation, p, expressed in terms of the percentage by which the user profile is to be extended, and two lists of indicative items: L_m is the list of indicative items for male users and L_f, is the list of indicative items for female users. The lists are created by training a logistic regression model on labeled training data (the same data that are to be obfuscated). The coefficients $\beta = \{\beta_0, \beta_1, \dots, \beta_M, \}$ of the logistic regression capture the extent to which each item is correlated with the class attribute gender. The coefficients are used to select the items for the two lists and order them according to the strength of the association. The higher the coefficient is, the more strongly the item is correlated with the attribute class. We extend user profiles by adding items until they are p percent longer than the original profile. When we are working with rating data (as opposed to implicit data), an added item receives either rating that is predicted for the user (using imputation, which is explained below) or average ratings.

Once an item has doubled its frequency with respect to the original data, it is no longer added. This mechanism is used by BlurM(or)e (Strucks et al., 2019), where it was shown to work well and, for this reason, adopted by PerBlur. We refer to this mechanism as "stop after doubled". The goal is to help to keep the overall distribution of items naturalistic. If "stop after doubled" is not applied, then the items in the top ranks of L_m and L_f will occur in a large number of user profiles, creating a "spike" in the item histogram. Such spikes make it obvious that the data set was obfuscated and conflict with our goal to design a system in which obfuscation is unobtrusive. Such a "stop after doubled" mechanism was not relevant in the original BlurMe (Weinsberg et al., 2012) work, since only 10% of the data was obfuscated at a time, so the items used for obfuscation would not be obvious in item histogram.

3.1. Standard PerBlur

Now we will discuss the specifics of PerBlur. We start with standard PerBlur, which is shown in Algorithm 1. First the algorithm creates personalized lists of indicative items (Lines 1–17). PerBlur is built on the insight that if the items added to the user profile for the purpose of obfuscation could have a close match to user preferences, then recommendation performance has a better chance of being maintained when the obfuscated data is used for training. To this end, PerBlur adds ratings (or interactions) to a user's profile that are consistent with the user's preferences, but are at the same time indicative for the opposite gender. Specifically, PerBlur uses a personalized list of indicative items for each user, $Personalized_L^u$. This list is created by intersecting a personalized list of preferred items for each user with the list of indicative items for the opposite gender (L_f for male users and L_m for female users). The personalized list is a list of items ranked in order of the probability that the user will have rated the item.

To create the personalized list, we need a recommender algorithm that imputes items (i.e., predicts ratings or interactions). In the case of rating data, this algorithm must produce a confidence score (and not just a rating prediction or a ranking score) since we are interested in the chance that a user will rate the item and not the user preference. We turn to the widely used user-based collaborative filtering algorithm (UserKNN). UserKNN predicts a rating for the target user u on a given item i by calculating the set of neighbors nearer than a specific distance threshold, θ, who have also rated this item. We choose UserKNN since the count of the neighbors used to make a prediction for an item is a straightforward choice of a confidence score. We rank the items by count from high to low to arrive at $List^u_{NCounts}$, our personalized list for each user. In order to make the item list effective for obfuscation, we do not use $List^u_{NCounts}$ directly. Rather, we create a final personalized item list ($Personalized_L^u$) for each user u by intersecting $List^u_{NCounts}$ with L_f (if u is male) or L_m (if u is female).

This approach runs risk that the final personalized item list $Personalized_L^u$ contains items that are not particularly specific to the opposite gender (because they are too far down the list L_f or L_m). For this reason, we impose a threshold on L_f and L_m. Note that BlurMe never reaches the bottom of L_f or L_m since it chooses the same items for all the users. PerBlur, however, reaches further down the list since it is attempting to leverage personal items. For this reason, the cutoff L_f and L_m is important for PerBlur, as our experiments will show.

Note that it is important to use an appropriate evaluation pipeline for assessing the performance of recommender systems on obfuscated data. We will discuss this point further in Section 6.1. We already state a key point here: imputation is trained and operates on training data only and never predict items in the test set being used to evaluate the recommender system.

3.2. PerBlur with removal

Next, we move to the second variant of PerBlur, namely "PerBlur with removal" shown in Algorithm 2. This algorithm takes data obfuscated by standard PerBlur as input, and removes items. Removal has two goals: First, it keeps the density of the obfuscated data close to the density of the original data. Removal is carried out so that the total number of user ratings (or user–item interactions) for each item remains close to the total number in the original data set. We spread out the items that need to be removed evenly over all users. For users with very short profiles, we do not remove items. In our experiments, we set the threshold defining very

Algorithm 1: Standard PerBlur

Input:
- p: percentage of obfuscation,
- users' binary gender information,
- L_f (L_m): list of indicative items for females (respectively males)
- Original user-item matrix $\mathcal{R}$ ($\mathbb{N}$ users, $\mathbb{M}$ items)
- Initial count: user profile size at time $t = 0$

Output: Standard PerBlur user-item matrix $\mathcal{R}'$ ($\mathbb{N}$ users, $\mathbb{M}$ items)

// 0. PerBlur Personalized lists of indicative items

1 Confidence score for recommendation based on UserKNN[2];
2 **for** *(user u in* $\mathbb{N}$*)* **do**
3   **for** *(item i in* $\mathbb{M}$*)* **do**
4   Similarity computation finds nearest neighbor candidates;
5   Sort selected items based on the number of possessed neighbor candidates;
6 $List^u_{NCounts}$ contains a list of counts for each user u;
7 **for** *(user u in* $\mathbb{N}$*)* **do**
8   Fix a cutoff on L_f and L_m
9   // we set the cutoff to *Top-50*, in the rest of the experiments
10   Create new personalized list of indicative items for u: $Personalized^u_L$;
11   **if** *(u is a Female)* **then**
12   // $Personalized^u_L = List^u_{NCounts} \cap L_m$
13   **for** *item* $i \in List^u_{NCounts}$ **do**
14             $Personalized^u_L = Personalized^u_L$. add ($i$) if item $i \ni L_m$
15   **else**
16   // $Personalized^u_L = List^u_{NCounts} \cap L_f$
17   Do the same steps (Line 9 to 12) but for a Male target user u
18 // 1. Obfuscation: adding extra ratings/interactions
19 **for** *(user u in* $\mathbb{N}$*)* **do**
20   count = initial count [u] $*$ p
21   added = 0
22   **while** *added* $<$ *count* **do**
23   i = picks the item in the first position in $Personalized^u_L$
24   **if** $\mathcal{R}'[u, i] == 0$ **then**
25             $\mathcal{R}'[u, i] = value$
26   added += 1
27   // For rating data, the rating value is either predicted using imputation or average ratings.
28   Total added += added

short profiles to 20, meaning that in the obfuscated data no user can have less than 20 interactions. Our exploratory experiments demonstrated that the success of obfuscation is not particularly sensitive to the threshold. Note that in standard PerBlur we keep track of the number of added items so that we can remove the same number later.

Second, removal contributes to the obfuscation. In other words, item removal can help to mask the gender of the user. Specifically, removing gender-indicative items in a controlled way, could potentially help to confuse the gender classifier, without unduly impacting recommendation performance. To this end, PerBlur proposes a new removal strategy, *greedy removal*, which removes items in the order of their indicativeness for the gender of the user whose profile is being obfuscated. The greedy removal strategy extends our previous work, BlurM(or)e, Strucks et al. (2019) which proposed random removal strategy. The random removal strategy chooses items for removal in a random manner.

When we evaluate the ability of obfuscation to block gender inference in Section 5, we apply greedy removal to BlurMe for comparison. We compare BlurMe with greedy removal to BlurMe with random removal. We see that removal helps to reduce gender inference and also that its contribution to recommendation lies in the area of improving diversity, discussed in Section 8.

4. Experimental setup

4.1. Data sets

We test our approach on three data sets. The first two are user–item matrices containing ratings (explicit feedback): MovieLens (Harper & Konstan, 2015) and Flixster (Zafarani & Liu, 2009). For MovieLens, we use the MovieLens 1 million (ML1M) release.

[2] We used $\theta = 0.6$ for MovieLens, $\theta = 0.45$ for Flixster, and $\theta = 0.4$ for LastFM.

Algorithm 2: PerBlur with removal

Input:
- Standard PerBlur user-item matrix $\mathcal{R}'$ ($\mathbb{N}$ users, $\mathbb{M}$ items)
- Removal mode = {Random, Greedy}
- Total added: total number of extra ratings (interactions) added
- Interaction count: user profile size after adding $p\%$ extra ratings/interactions.
- Removal threshold

Output: PerBlur with removal user-item matrix $\mathcal{R}''$ ($\mathbb{N}$ users, $\mathbb{M}$ items)

```
1  // 2. Obfuscation: Removing certain items
2  for user u in N do
3      if Interaction count ≥ Removal threshold then
4          // Removal threshold is chosen by us to be 20.
5          remove count += 1
6  To be removed = Total added / remove count
7  // To be removed: contains the number of ratings (or interactions) that will be removed from
   individual user profiles.
8  for user u in N do
9      if (Interaction count ≥ Removal threshold) then
10         if (u is a Female) then
11             removed = 0
12             while (removed < To be removed [u]) do
13                 i = picks an item from L_f
14                 // i depends on the removal mode: random or greedy
15                 if R'[u, i]! = 0 then
16                     R''[u, i] = 0
17                     removed += 1
18                 // R'' is R' after applying the removal
19         else
20             Do the same steps (Line 5 to 13) but for a Male target user u
```

Table 3
Summary of data sets.

Data sets	#Users	#Items	#Ratings	#Sparsity (%)	Gender (F/M)
ML1M	6040	3706	1000209	4.47	1709/4331
Flixster	2372	2835	369059	5.49	1480/890
LastFM	884	55686	655929	0.01	382/502

For Flixster, we select users with at least 15 ratings and movies with at least 20 ratings, which results in a subset of ratings for 2.8K items by 2.4K users. The ratings are between $[1, 5]$ for both data sets. The third is a user–item matrix containing interactions (implicit feedback): LastFM data (Bertin-Mahieux, Ellis, Whitman, & Lamere, 2011). We use artists as the items. Our experimental data set contains users who listened to at least 10 artists and artists to which at least 10 users have listened. The result is a subset of 884 users and 56K artists. The three data sets contain binary information on user gender, i.e., the gender of a user can be either male or female. We choose these data sets because they contain gender information and because they are publicly available, for reproducibility purposes. Table 3 summarizes the statistics of the data sets that we used. In can be seen that the MovieLens and LastFM data sets are quite sparse (4.47% and 0.01%, respectively), and the Flixster data set is somewhat less sparse (5.49%). Note also that in ML1M and LastFM, there are more male than female users (ML1M: 72% male vs. 28% female and LastFM: 57% male vs. 43% female), but in Flixster there are more female than male users (38% male vs. 62% female).

4.2. Evaluation metrics

The evaluation that we carry out in this paper measures four different aspects of the obfuscated data: (1) the *success of the obfuscation* (2) how well *recommender system performance* is maintained on obfuscated data, (3) how obfuscation impacts *fairness* by making the difference in the quality of the recommendations between males and females larger, and (4) how obfuscation impacts *diversity* of recommended items with respect to gender-stereotypicality. In this section, we present the metrics that we use for each of these aspects.

4.2.1. Success of obfuscation

For gender inference, we compute the Area Under the Curve (AUC) using the mean Receiver Operating Characteristic (ROC) curve computed across ten folds. For the ROC, the true positive rate (TPR or sensitivity) is calculated as the rate of correctly classified male users out of males in the data set and the false positive rate (FPR) is calculated as the rate of users incorrectly classified as male out of females in the data set. The ROC curve is plotted with TPR against the FPR where TPR is on y-axis and FPR on the x-axis. We consider gender obfuscation to be successful when the prediction accuracy is close to the average accuracy of random guessing, i.e., 0.5 , which means that the classifier does not have the ability to separate the classes.

4.2.2. Recommender system performance

In our experiments, we compare recommenders trained on the original data with recommenders trained on data obfuscated with different variants of BlurMe and PerBlur. We carry out rating prediction for comparison with previous work, but our main focus is on Top-N recommendation. The goal is to keep the performance of recommender systems trained on the obfuscated data as close as possible to recommender systems trained on the original data.

Here, we define the metrics that we use. For rating prediction we use mean absolute error (MAE), the mean of the absolute difference between each prediction and rating for all the ratings of users in the test set. If there are n held-out ratings in the test set, the MAE is computed as follows:

$$\text{MAE} = \frac{1}{n}\sum_{u,i} |p_{u,i} - r_{u,i}|$$

where $p_{u,i}$ is the predicted rating for user u on item i and $r_{u,i}$ is the rating value of user u on item i in the test set.

For top-N recommendation, we use Hit Ratio@10 and Top10.nDCG. To compute Hit Ratio@10 (HR@10), we consider an item a "hit" if it is relevant and is ranked as one of the top-N (N = 10) items that we recommend. In the case of the rating data being used as implicit data, we threshold at 3.5, which means any predicted rating above 3.5 will be considered as relevant (= 1) and below will not be relevant (= 0). HR@10 is defined as the count of hits ($\#Hits$) divided by the total user–item pairs in the test set ($\#counts$).

$$\text{HR} = \frac{\#Hits}{\#counts}$$

Note that in order to give a ranking perspective to our rating experiments, we rank according to rating prediction and calculate HR@10, although this method would not be used to generated Top-N recommendations in a practical setting.

Normalized Discounted Cumulative Gain (Top10.nDCG) measures the utility that a user is expected to obtain from a recommender based on that user's estimated utility for individual items and the position in the list at which those items were presented (Ekstrand et al., 2018). In order to compute nDCG, first we truncate the recommendation list to 10. Then, we compute the discounted cumulative gain (DCG) of the recommended order and the DCG of the ideal order (iDCG). The $DCG_{L_{Rec},u}$ is defined as:

$$DCG_{L_{Rec},u} = \mu_u(l_1) + \sum_{i=2}^{|L_{Rec}|} \frac{\mu_u(l_i)}{log_2 i}$$

where l_i is the ith item in the recommendation list L_{Rec} and $\mu_u(l_i)$ is user's u utility for item l_i. We define $\mu_u(l_i)$ as a binary utility: if a user u consumed item i then, $\mu_u(l_i) = 1$. Otherwise, items for which no data is available are assumed to have a utility of 0. Then, the $nDCG_{L_{Rec},u}$ for a recommendation list L_{Rec} generated for a target user u is the ratio of DCG of recommended order ($DCG_{L_{Rec},u}$) to DCG of ideal order ($iDCG_u$).

$$nDCG_{L_{Rec},u} = \frac{DCG_{L_{Rec}}, u}{iDCG_u}$$

Note that for the rating data, nDCG is also calculated with respect to thresholded ground truth.

To illustrate the impact of obfuscated data on recommendation performance, we report the gain (+) or drop (−) of the recommender performance on obfuscated data with respect to the recommender performance on original data.

4.2.3. Fairness

Recall that we study fairness in terms of the ability of the system to provide good recommendations for both females and males. To measure fairness, we split users in the test set into female users and male users. Then, we measure $nDCG_F$ and HR_F for female test users and $nDCG_M$ and HR_M for male test users to illustrate the overall satisfaction obtained by each gender group and the difference between them.

For each gender, we report the gain (+) or drop (−) of the recommender performance on obfuscated data. We also report the absolute magnitude of the difference between the male users' drop and the female users' drop. If an obfuscation strategy is fair, this difference should remain as small as possible. In cases where the obfuscation strategy increases the performance of the recommender system, this difference should also remain small, but it is not as important as in cases where performance is lost.

4.2.4. Diversity

We are interested in keeping the number of gender-stereotypical items that the recommender system recommends to users under control. Our study of diversity, for this reason, is focused on the proportion of correctly recommended items that are gender-stereotypical. In this work, we define a gender-stereotypical recommendation as an item that is highly typical for a particular gender.

We calculate the number of user–item pairs for which the item is correctly recommended to user u and is also considered a highly typical item. For this purpose, we use the top {10, 20, 50} items of the highly indicative items list L_m (if u is a male user) or L_f (if u is a female user).

Table 4
Gender inference results measured in terms of AUC using logistic regression and SVM classifiers on: **original** ML1M, Flixster and LastFM data sets. The ± represents the standard deviations of the results over different ten folds.

	AUC	
	Logistic regression	*SVM*
ML1M	0.87 ± 0.02	0.82 ± 0.04
Flixster	0.87 ± 0.02	0.81 ± 0.04
LastFM	0.77 ± 0.06	0.72 ± 0.04

4.3. Algorithms and evaluation setup

In this section, we describe the recommender system algorithms and gender inference algorithms that we will use in our experiments.

4.3.1. Gender inference algorithm

For gender inference, we choose logistic regression because it is mentioned in literature, Weinsberg et al. (2012) and Chen et al. (2014), as the best performing classifier for gender inference on recommender system data. This was confirmed by our exploratory experiments. We also report results here on SVM, which was the second strongest classifier in our exploratory experiments. We apply normalization (L^2-norm)[3] to the user–item matrix to scale all ratings to values in [0, 1].

To evaluate gender inference, we carry out ten fold cross-validation (using StratifiedKFold[4]). In every iteration, we train the classifier on 9 folds and we test on the 10th fold. Hyperparameters are selected from the training set with grid search (GridSearchCV[5] from Sklearn). The test results for the classifier are reported in Table 4 in terms of AUC. Our goal will be to reduce these scores. Recall that we consider gender inference to have been successfully blocked once AUC has been reduced to the level of a random classifier 0.5. Scores below 0.5 show that the more we add extra ratings (interactions), the lower the inference score is. However, scores lower than 0.5 are not ideal cases of blocking because they could be flipped.

4.3.2. Recommender algorithms

For our recommendation experiments, we use two state-of-the-art algorithms commonly used in collaborative filtering recommender systems: ALS (Pilászy, Zibriczky, & Tikk, 2010) and BPRMF (Rendle, Freudenthaler, Gantner, & Schmidt-Thieme, 2009). ALS is a matrix factorization model trained with alternating least squares. We choose this algorithm because it takes a user–item matrix containing ratings (explicit data) as input. BPRMF is a matrix factorization model trained using the Bayesian Personalized Ranking from implicit data. BPRMF is a learning-to-rank algorithm that optimizes pairwise ranking. This algorithm takes a user–item matrix containing interactions (implicit data) as input. We use the implementations of the Lenskit Python (lkpy) toolkit (Ekstrand, 2020).

To evaluate recommender performance, we randomly sample (without replacement) 80% of the items in each user profile as our training set and 20% as our test set. The hyperparameters for each algorithm (the number of features and the number of iterations for ALS and the number of epochs, batch size and features for BPRMF) are tuned using cross validation on the training set. We evaluate the performance of the recommender system algorithm using a special adaptation of 1+random, which is explained in Section 6.1.

5. Blocking of gender inference

In this section, we report experimental results that demonstrate the ability of gender obfuscation to block gender inference. Table 5 presents the performance of the gender classifier on data obfuscated with different variants of BlurMe and Table 6 presents different variants of PerBlur. The classifier is trained on the original data, and tested on obfuscated data. We test four levels of obfuscation, corresponding to adding 1%, 2%, 5%, and 10% extra ratings/interactions to each user profile in the original data. Recall that the indicative items lists L_f and L_m used by BlurMe and PerBlur are selected using logistic regression. Here, we evaluate classification results with respect to that same logistic regression classifier. We also test an SVM in order to confirm that the gender obfuscation transfers to a classifier not used for the selection of the indicative item lists.

Tables 5 and 6 show that when data is obfuscated with any of the obfuscation approaches, classification performance is lower than on the original data (cf. Table 4). Recall that lower classification performance is our goal, since it represents improved user privacy. We can see the impact that obfuscation has on lowering the classification performance is evident for both logistic regression and SVM, confirming that our obfuscation approach is not specific to the logistic regression classifier.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.normalize.html.
[4] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html.
[5] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

Table 5
Gender inference results measured in terms of AUC for different BlurMe obfuscations (with no removal, random removal, and greedy removal) on ML1M, Flixster, and LastFM data sets. The ± are standard deviations of the results over ten folds.

Gender inference		Obfuscation strategies		**Logistic regression**				**SVM**			
				Extra ratings/interactions				*Extra ratings/interactions*			
Data sets		**Personalization**	**Removal**	*1%*	*2%*	*5%*	*10%*	*1%*	*2%*	*5%*	*10%*
ML1M	*BlurMe*	*None*	*No removal*	0.76 ± 0.03	0.69 ± 0.03	0.48 ± 0.05	0.22 ± 0.06	0.74 ± 0.03	0.67 ± 0.03	0.42 ± 0.06	0.16 ± 0.06
			Random	0.75 ± 0.03	0.69 ± 0.03	0.43 ± 0.05	0.13 ± 0.05	0.74 ± 0.03	0.66 ± 0.02	0.38 ± 0.08	0.10 ± 0.04
			Greedy	0.59 ± 0.03	0.49 ± 0.03	0.22 ± 0.04	0.05 ± 0.02	0.53 ± 0.03	0.42 ± 0.03	0.14 ± 0.04	0.02 ± 0.01
Flixster	*BlurMe*	*None*	*No removal*	0.65 ± 0.05	0.59 ± 0.05	0.41 ± 0.05	0.19 ± 0.04	0.62 ± 0.05	0.55 ± 0.05	0.35 ± 0.05	0.14 ± 0.04
			Random	0.65 ± 0.05	0.59 ± 0.05	0.38 ± 0.05	0.14 ± 0.03	0.62 ± 0.05	0.55 ± 0.05	0.32 ± 0.05	0.10 ± 0.03
			Greedy	0.44 ± 0.06	0.33 ± 0.05	0.17 ± 0.03	0.06 ± 0.02	0.39 ± 0.06	0.28 ± 0.05	0.13 ± 0.03	0.04 ± 0.02
LastFM	*BlurMe*	*None*	*No removal*	0.66 ± 0.06	0.57 ± 0.07	0.35 ± 0.07	0.16 ± 0.05	0.61 ± 0.07	0.45 ± 0.08	0.15 ± 0.05	0.04 ± 0.02
			Random	0.65 ± 0.06	0.55 ± 0.07	0.27 ± 0.07	0.03 ± 0.02	0.60 ± 0.07	0.43 ± 0.08	0.08 ± 0.04	0.006 ± 0.002
			Greedy	0.52 ± 0.07	0.39 ± 0.07	0.17 ± 0.05	0.05 ± 0.02	0.39 ± 0.07	0.21 ± 0.05	0.03 ± 0.02	0.007 ± 0.003

Comparing BlurMe results in Table 5, we see that BlurMe with greedy removal outperforms the other approaches of BlurMe with no removal and BlurMe with random removal. BlurMe with greedy removal requires less obfuscation (fewer extra ratings/interactions) to bring the performance of the classifier close to 0.5 AUC. This is due to the fact that BlurMe with greedy removal uses our proposed removal strategy which removes items in the order of their gender indicativeness. Also, we observe that BlurMe with no removal and BlurMe with random removal perform similarly and both require heavier obfuscation, and, as such, can be considered less effective than BlurMe with greedy removal. For this reason, next in Table 6, we omit PerBlur with random removal and we continue with PerBlur with no removal and PerBlur with greedy removal.

Table 6 shows the gender inference results on PerBlur data for the case of *no removal* and the case of *greedy removal* with different personalization cutoffs. The "personalization" column gives the length at which $Personalized_L^u$ is truncated (see Line 7 to Line 14 of Algorithm 1). We recall that the personalization proposed by PerBlur attempts to create a personalized list of indicative items that are close to the user preferences. In Table 6, we see that PerBlur with *no removal* succeeds to lower the gender inference score but with more obfuscation (more extra ratings/interactions). We note that we tested the case in which there is no threshold for the personalization (we call it "All items"). We do not include this results in the table, but instead mention that we found that the inference of the classifier is at the same level as it is for the original data. This is to be expected because without the threshold there is no influence from the indicative item list.

We observe in Table 6 that PerBlur with greedy removal outperforms the other variants of PerBlur with no removal, since it can bring the performance of the classifier close to 0.5 AUC with less obfuscation (fewer extra ratings/interactions). This demonstrates the importance of using greedy removal strategy which removes items in the order of their gender indicativeness. We consider gender obfuscation to be successful at levels of obfuscation at which AUC is close to 0.5. As we previously mentioned, levels of classification performance lower than 0.5 actually reveal the gender because the point reliably in the opposite direction. Among the personalization levels, Top-50, Top-100 from $Personalized_L^u$, Top-50 items performs consistently well, and we adopt this setting for the PerBlur experiments in the rest of the paper.

It is important to remember that the ability of gender obfuscation to block the SVM classifier seen in Tables 5 and 6 is a demonstration of the transferability of our approach. Recall from Section 3 that the indicative items lists are chosen using logistic regression. It makes sense, then, that adding these items in order to obfuscate data would be able to prevent the classifier from making accurate predictions. The SVM results assure us that the items chosen using logistic regression actually have a general blocking power, since using these items to obfuscate data is also able to block the ability of the classifier to make predictions.

A different view on the gender inference performance is presented in the ROC curves in Fig. 2. Here, we show ML1M, and leave out the other data sets since the pattern is similar. These curves dramatically show the level of obfuscation (extra ratings or interactions) at which the performance of the classifier collapses (i.e., the performance approaches the diagonal). On the basis of these curves, we choose the levels of obfuscation for each obfuscation approach that we will investigate for each data set in the remainder of the paper. These settings constitute a "rough and ready" operating point at which we know that the gender prediction performance has collapsed. Specifically, for the ML1M data set, we add 5% extra ratings/interactions to BlurMe with *no removal* and Standard PerBlur, and we add 2% extra ratings/interactions for PerBlur with *greedy removal* (see Fig. 2). For the Flixster data set, we add 2% extra ratings/interactions to BlurMe with *no removal*, 10% to Standard PerBlur, and we add 2% extra ratings to

Table 6
Gender inference results measured in terms of AUC on PerBlur (No removal and greedy removal), for ML1M, Flixster, and LastFM data sets. The ± are standard deviations of the results over ten folds. We note that for rating data, PerBlur with average ratings has quite similar results to PerBlur with predicted ratings. For this reason, we only report results of PerBlur with predicted ratings.

Gender inference		Obfuscation strategies		**Logistic Regression**				**SVM**			
				Extra ratings/interactions				*Extra ratings/interactions*			
Data Sets		**Personalization**	**Removal**	*1%*	*2%*	*5%*	*10%*	*1%*	*2%*	*5%*	*10%*
ML1M	*Standard PerBlur*	*Top-50 Items*	*No Remval*	0.80 ± 0.02	0.76 ± 0.03	0.59 ± 0.03	0.43 ± 0.09	0.78 ± 0.03	0.73 ± 0.03	0.52 ± 0.02	0.34 ± 0.12
		Top-100 Items		0.81 ± 0.02	0.78 ± 0.03	0.64 ± 0.04	0.39 ± 0.03	0.79 ± 0.03	0.75 ± 0.03	0.55 ± 0.04	0.28 ± 0.03
	PerBlur with removal	*Top-50 Items*	*Greedy*	0.66 ± 0.03	0.53 ± 0.03	0.26 ± 0.03	0.14 ± 0.05	0.61 ± 0.03	0.46 ± 0.03	0.17 ± 0.02	0.09 ± 0.05
		Top-100 Items		0.68 ± 0.03	0.56 ± 0.04	0.26 ± 0.04	0.10 ± 0.02	0.63 ± 0.03	0.48 ± 0.03	0.17 ± 0.03	0.07 ± 0.01
Flixster	*Standard PerBlur*	*Top-50 Items*	*No removal*	0.78 ± 0.04	0.75 ± 0.04	0.67 ± 0.04	0.57 ± 0.06	0.73 ± 0.04	0.69 ± 0.04	0.57 ± 0.04	0.45 ± 0.08
		Top-100 Items		0.79 ± 0.04	0.77 ± 0.04	0.69 ± 0.04	0.55 ± 0.05	0.75 ± 0.04	0.72 ± 0.04	0.59 ± 0.05	0.41 ± 0.04
	PerBlur with removal	*Top-50 Items*	*Greedy*	0.56 ± 0.05	0.48 ± 0.04	0.27 ± 0.03	0.13 ± 0.02	0.56 ± 0.05	0.42 ± 0.04	0.22 ± 0.03	0.10 ± 0.02
		Top-100 Items		0.57 ± 0.05	0.43 ± 0.05	0.19 ± 0.04	0.08 ± 0.02	0.47 ± 0.05	0.37 ± 0.06	0.14 ± 0.04	0.06 ± 0.02
LastFM	*Standard PerBlur*	*Top-50 Items*	*No removal*	0.70 ± 0.06	0.63 ± 0.07	0.49 ± 0.08	0.42 ± 0.14	0.68 ± 0.06	0.56 ± 0.07	0.34 ± 0.09	0.28 ± 0.15
		Top-100 Items		0.71 ± 0.06	0.64 ± 0.07	0.44 ± 0.06	0.28 ± 0.09	0.69 ± 0.06	0.58 ± 0.07	0.27 ± 0.05	0.13 ± 0.09
	PerBlur with removal	*Top-50 Items*	*Greedy*	0.53 ± 0.06	0.39 ± 0.06	0.21 ± 0.06	0.17 ± 0.09	0.39 ± 0.06	0.21 ± 0.04	0.07 ± 0.03	0.06 ± 0.06
		Top-100 Items		0.54 ± 0.07	0.36 ± 0.07	0.12 ± 0.04	0.06 ± 0.04	0.42 ± 0.06	0.18 ± 0.03	0.02 ± 0.01	0.02 ± 0.02

PerBlur with *greedy removal*. For the LastFM data set, we add 2% extra ratings/interactions to BlurMe with *no removal*, and 5% extra ratings/interactions to Standard PerBlur, and we add 1% extra ratings/interactions to PerBlur with *greedy removal*.

6. Recommendation performance

Now that we have established the effectiveness of PerBlur in blocking gender inference, we turn to the evaluation of its ability to maintain recommendation prediction performance.

6.1. Evaluation procedure

In order to evaluate obfuscated data, it is necessary to have an evaluation procedure that creates a fair environment to compare top-N recommendation performance between different obfuscation techniques. Because obfuscation adds and subtracts ratings (or interactions), designing a procedure is non-trivial. Unless specific attention is paid to how training and test splits are created, the addition and subtraction of ratings (or interactions) to the user profiles will lead to the test set being different for the different versions of the data that are being compared with each other. The result would be that the test conditions are no longer directly comparable. For example, a particular condition might add easy-to-predict and remove difficult-to-predict ratings, meaning that the prediction score no longer reflects that performance of the recommender algorithm.

We introduce a new evaluation procedure for obfuscated data that ensures that different conditions are comparable. Our procedure works as follows. We randomly sample 80% of each user profile for the training set and we keep the remaining 20% for the test set. The choice of static splitting plays a key role in preventing obfuscation from adding items into the test set. Obfuscation is applied only to the training set. The effect is that the test set remains connected to users in the training set, and will remain the same across all of the conditions.

For Top-N recommendation, the procedure needs to address an additional challenge. Specifically, we would like to be able to use the *1+random* protocol (Bellogin, Castells, & Cantador, 2011; Cremonesi, Koren, & Turrin, 2010) and still maintain a fair comparison across conditions. Under this protocol, a test item is added to a set of random candidate items (here, 1000 items) that are drawn from a set of possible candidate items. In order to maintain fairness in our evaluation procedure, we must look not only at the test set, but also at the set of possible candidate items from which the random items are drawn. The core of the challenge is the following. If *1+random* makes use of a candidate set drawn from the training data, that candidate set will change from condition

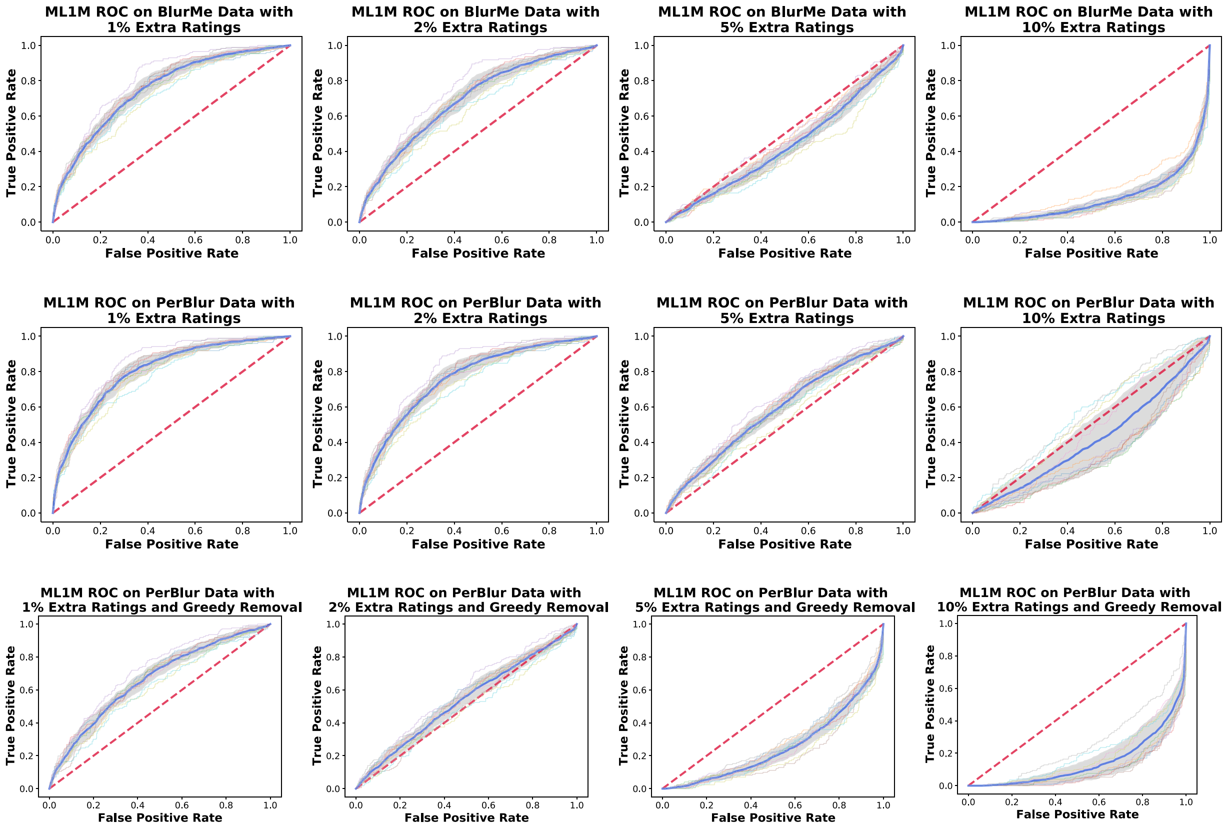

Fig. 2. ROC AUC of a logistic regression classifier on BlurMe (with no removal), Standard PerBlur (with no removal) and PerBlur with removal (greedy removal) for different degrees of obfuscation (1%, 2%, 5% and 10%) for **ML1M** Data. PerBlur data is created with Top-50 personalized list of indicative items and using predicted ratings. We observe that BlurMe and PerBlur with *no removal* require 5% extra ratings/interactions to perform like a random classifier. PerBlur with *greedy removal* requires only 2% of extra ratings/interactions.

to condition, since each type of data obfuscation makes different additions (and subtractions) to the data. When obfuscation adds and deletes items, it will impact the candidate set. Specifically, it will change the set of items that compete with the relevant item for each user across conditions. This issue does not occur with data that is not obfuscated.

To address this problem, we adapt *1+random* evaluation for use with obfuscated data. For each user, we define a candidate set of $C = 1000$ items. In Fig. 3, we describe the process of generating these candidate items. The items must be selected among items that are *not* rated (= ?) by the user. To ensure that these items are comparable across conditions, we create a large set of possible candidate items for each user by intersecting the items that are *not* rated by the user in the original training set as well as *not* rated by the user in all the obfuscated training sets. We then draw C random items to create the candidate set for each user from this set of possible candidate items. Each relevant item in the set of a user's test items is injected in turn into the user's candidate set, and then recommendation is performed and the ranking metric is calculated. We adopt this evaluation procedure for the comparison of recommendation performance carried out in the next section.

Note that this procedure requires experiments to be planned carefully in advance. Building the candidate set requires an intersection involving data from all conditions that are being compared. It is not possible to compare two types of obfuscated data, and then add a third type later because the candidate set must necessarily change.

It is important to understand why building the candidate set from the test items of the other users is not a viable solution. For this methodology (Bellogin et al., 2011), the candidate lists include, for all users, all the items having a test rating/interaction by some user and no training rating/interaction by the target user. When 1+random uses candidates drawn from the test items of other users, the lists have to exclude items that are in the training set of the target user's profile. The training part of the profile is exactly the part that changes from one type of obfuscation to the other. Again, we see that the candidate list will change across comparative conditions. For this reason, we build the candidate list for a given target user using items that are in the training data of all of the obfuscated data sets being compared, but are not rated by (or interacted with) the target user and are not in the test set of the target user.

It is very important to remember that only the scores of conditions that contribute to building the candidate sets can be directly compared. In order to understand this point in more detail, consider the impact that different types of obfuscation have on the candidate sets. Recall that the items added by obfuscation to the training data will not occur in the users' individual candidate sets.

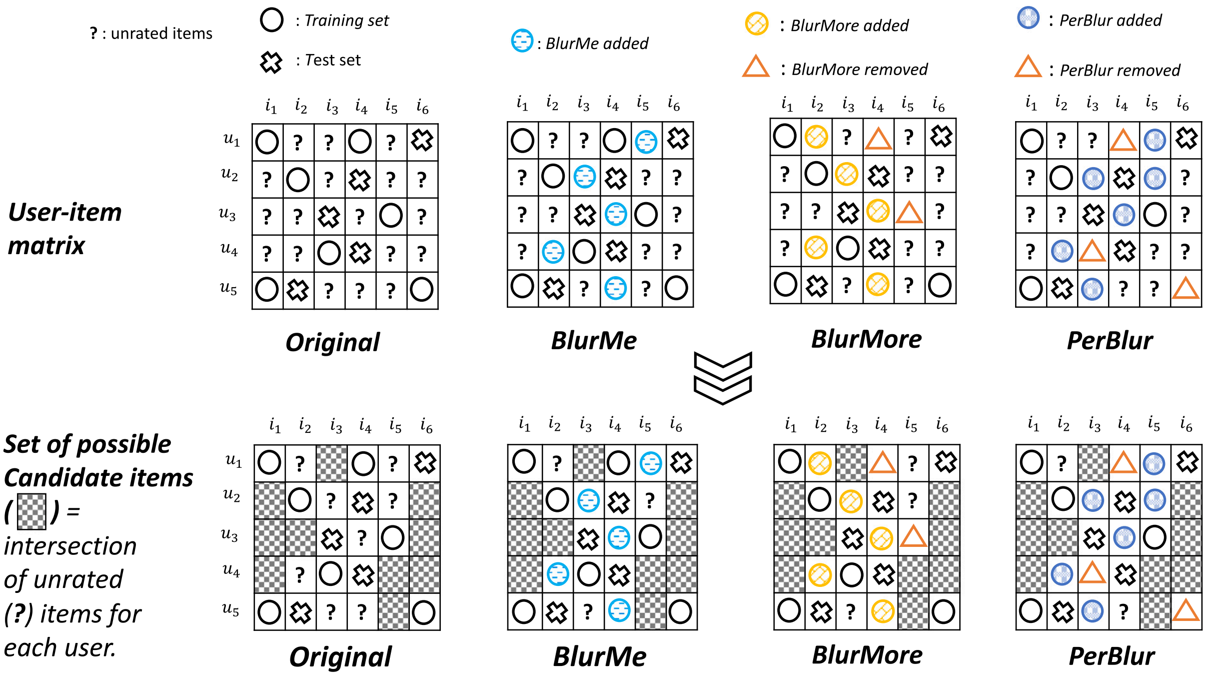

Fig. 3. Generation of possible candidate items: the intersection of unrated items from the original training set, original test set and the training sets of different obfuscation conditions.

Consequently, if highly personalized obfuscation approaches are compared, then users' candidate sets may contain less personalized items. Such candidate sets could offer less competition with the relevant item that is being tested in 1+random evaluation, leading to higher absolute scores. The opposite could be true if less personalized obfuscation is used. In short, scores can only be compared relatively within a set of conditions that all contributed to the candidate sets.

6.2. Comparing recommendation performance

In this section, we compare recommendation performance in order to measure the extent to which the accuracy of a recommender is impacted when it is trained on obfuscated data. We use the evaluation procedure just described. The training/test split is kept constant. In order to understand the range of variability, we repeat the evaluation five times. Each repetition involves the selection of a new candidate set for each user. We report the average and standard deviation of the repetitions. Recall that we are testing obfuscation levels that we have previously determined to lead to collapse of the predictive ability of the gender classifier.

First, we look at the results of the ALS algorithm, which takes rating data as input and gives rating predictions as output. Remember that our main focus is Top-N recommendation, but we test rating prediction because that was the focus of previous work, most importantly Weinsberg et al. (2012). The results are shown in Table 7. We report rating prediction with MAE. We also rank items by their predicted ratings, which allows us to get a top-N view of ALS and ranking prediction. The performance of this ranking is reported as HR@10.

The main insight gained from Table 7 is that it is possible to train a recommender on obfuscated data, and still maintain a comparable performance level as is achieved when the recommender is trained on the original, unobfuscated data. This conclusion is consistent with the BlurMe (Weinsberg et al., 2012) rating prediction experiments. Recall however, that in contrast to Weinsberg et al. (2012), we obfuscate the full data set, rather than just 10%. We find that the performance in the case of obfuscated data can actually exceed the performance in the case of the original data (i.e., MAE falls below the level of Original). This can be explained by the fact that in some cases, extending the profile gives a boost. We see that in terms of MAE, PerBlur and BlurMe achieve approximately the same performance level, and both outperform the original. In terms of HR@10 it remains close. In real-world application scenarios, we are not particularly interested in rating prediction, nor would we choose to carry out top-N recommendation by ranking on the basis of predicted ratings. However, these experiments serve to give insight into how gender obfuscation works, and link our analysis of PerBlur to the related work on BlurMe, which studied rating prediction.

Next, we look at BPRMF algorithm, which takes implicit data as input and gives a ranked list of items as output. Recall that ML1M and Flixster are binarized via thresholding and LastFM is interaction data, which is originally implicit. The results are shown in Table 8. We report TopN recommendation results measured with Top10.nDCG and HR@10.

We see in Table 8 that when data obfuscated with PerBlur is used, the recommendation performance comes very close to what is achieved on the original data for both Top10.nDCG and HR@10. This observation stands in contrast to the conventional wisdom

Table 7
Rating prediction results measured in terms of MAE and HR@10 using ALS on Original, BlurMe, and standard PerBlur Data (personalization with Top-50 indicative items) for ML1M and Flixster data sets. The scores report the average over five repetitions of the evaluation. The standard deviation for HR@10 is around 0.0001.

ALS		*Obfuscation strategies*		*MAE*	*HR@10*
Data sets		**Obfuscation level**	**Personalization**		
ML1M	*Original*	*0%*	*None*	0.7534	0.0125
	BlurMe	*5%*	*None*	0.7477	0.0126
	Standard PerBlur	*5%*	*Personalized with Average Ratings*	0.7485	0.0129
		5%	*Personalized with Predicted Ratings*	0.7497	0.0125
Flixster	*Original*	*0%*	*None*	0.7584	0.0071
	BlurMe	*5%*	*None*	0.7400	0.0070
	Standard PerBlur	*10%*	*Personalized with Average Ratings*	0.7415	0.0066
		10%	*Personalized with Predicted Ratings*	0.7410	0.0069

Table 8
Ranking prediction results measured in terms of Top10.nDCG (Δ wrt original Top10.nDCG) and HR@10 (Δ wrt original HR@10) using BPRMF on Original, BlurMe (no removal), and Standard PerBlur Data (personalization with Top-50 indicative items and no removal). The scores report the average over five repetitions of the evaluation. The standard deviation of Top10.nDCG and HR@10 is around 0.001 on ML1M and Flixster data sets. The standard deviation of Top10.nDCG and HR@10 is around 0.005 on LastFM data.

BPRMF		*Obfuscation strategies*		***nDCG*** *(Δ wrt original)*	***HR@10*** *(Δ wrt Original)*
Data sets		**Obfuscation level**	**Personalization**		
ML1M	*Original*	0%	*None*	0.1634	0.1712
	BlurMe	5%	*None*	0.1536 (−0.0098)	0.1633 (−0.0080)
	Standard PerBlur	5%	*Personalized with Average Ratings*	0.1603 (−0.0031)	0.1675 (−0.0037)
		5%	*Personalized with Predicted Ratings*	**0.1637** (+0.0003)	**0.1704** (−0.0009)
Flixster	*Original*	0%	*None*	0.1139	0.0628
	BlurMe	5%	None	0.1066 (−0.0073)	**0.0605** (−0.0023)
	Standard PerBlur	10%	*Personalized with Average Ratings*	0.1028 (−0.0112)	0.0602 (−0.0026)
		10%	*Personalized with Predicted Ratings*	**0.1099** (−0.0041)	0.0595 (−0.0033)
LastFM	*Original*	0%	*None*	0.0782	0.0603
	BlurMe	2%	*None*	**0.0839** (+0.0056)	**0.0722** (+0.0119)
	Standard PerBlur	5%	*Personalized with Interactions*	0.0752 (−0.0030)	0.0690 (+0.0087)

that privacy comes at the price of decreased recommendation performance. Recall that the obfuscation levels used here are chosen because they collapse the AUC curve to a random classifier. In other words, obfuscation defeats the classifier with a very small decrease in recommendation performance, if there is a decrease at all. Overall, PerBlur approaches the original performance more closely and more consistently than BlurMe.

Further in Table 8 we see that PerBlur that uses predicted ratings outperforms PerBlur that uses average ratings. This point is interesting since predicted ratings were not found to be particularly helpful by Weinsberg et al. (2012).

7. Maintaining fairness

Next, we move to investigate the impact of gender obfuscation on fairness. Here, we are concerned about the extent to which recommender algorithms trained on obfuscated data are able to maintain fairness for both genders. We investigate fairness by comparing the ranking prediction results calculated separately for males and females. These results are presented in Table 9.

Table 9
Ranking prediction results measured in terms of Top10.nDCG and HR@10 using BPRMF for female and male users. The standard deviation of Top10.nDCG and HR@10 for female and male users is around 0.001 on ML1M and Flixster data sets. The standard deviation of Top10.nDCG and HR@10 for female and male users is around 0.005 on LastFM data. $|\Delta_{nDCG_F} - \Delta_{nDCG_M}|$ measures the difference with respect to the original Top10.nDCG for both genders. $|\Delta_{HR_F} - \Delta_{HR_M}|$ measures the difference with respect to the original HR for both genders. The scores report the average over five repetitions of the evaluation.

BPRMF		*Obfuscation strategies*		$nDCG_F$ (*Δ wrt Original*)	$nDCG_M$ (*Δ wrt Original*)	$\lvert\Delta_{nDCG_F} - \Delta_{nDCG_M}\rvert$	HR_F (*Δ wrt Original*)	HR_M (*Δ wrt Original*)	$\lvert\Delta_{HR_F} - \Delta_{HR_M}\rvert$
Data sets		**Obfuscation Level**	**Personalization**						
ML1M	*Original*	0%	*None*	0.1478	0.1695	0.0000	0.1575	0.1768	0.0000
	BlurMe	5%	*None*	0.1338 (−0.0140)	0.1614 (−0.0082)	0.0058	0.1474 (−0.0101)	0.1697 (−0.0071)	0.0029
	Standard PerBlur	5%	*Personalized with Average Ratings*	0.1398 (−0.0080)	0.1684 (−0.0011)	0.0069	0.1489 (−0.0087)	0.1751 (−0.0017)	0.0069
		5%	*Personalized with Predicted Ratings*	**0.1435** (−0.0043)	**0.1716** (+0.0021)	0.0064	**0.1518** (−0.0057)	**0.1779** (+0.0011)	0.0068
Flixster	*Original*	0%	*None*	0.1115	0.1179	0.0000	0.0605	0.0671	0.0000
	BlurMe	5%	*None*	0.1060 (−0.0055)	0.1077 (−0.0103)	0.0048	0.0591 (−0.0014)	0.0630 (−0.0041)	0.0027
	Standard PerBlur	10%	*Personalized with Average Ratings*	0.1042 (−0.0074)	0.1004 (−0.0175)	0.0101	**0.0590** (−0.0015)	0.0624 (−0.0047)	0.0032
		10%	*Personalized with Predicted Ratings*	**0.1079** (−0.0037)	**0.1132** (−0.0047)	0.0010	0.0576 (−0.0029)	**0.0631** (−0.0040)	0.0011
LastFM	*Original*	0%	*None*	0.1052	0.0570	0.0000	0.0805	0.0445	0.0000
	BlurMe	2%	None	**0.1092** (+0.0040)	**0.0639** (+0.0069)	0.0029	0.0836 (+0.0031)	**0.0633** (+0.0188)	0.0157
	Standard PerBlur	5%	*Personalized with Interactions*	0.1033 (−0.0019)	0.0532 (−0.0038)	0.0019	**0.0873** (+0.0068)	0.0547 (+0.0102)	0.0034

The first point to notice in Table 9 is that all of our data sets have a gap between the performance for the two genders (see the first line of each section of the table, reporting results on the original data). For ML1M and Flixster, the systems perform better for males and for LastFM the systems perform better for females. A gender performance gap has been observed in many systems in the literature, e.g., Ekstrand et al. (2018).

We have previously seen that obfuscation sometimes improves recommendation, but often causes a small drop. Here, we see that the drop is not evenly distributed over both genders. Rather, one gender drops further than the other. The implication is that when obfuscating it is necessary to check that the recommender system performance is not impacted asymmetrically between the genders. This observation is new, and has not been previously reported in the literature.

In the columns $|\Delta_{nDCG_F} - \Delta_{nDCG_M}|$ and $|\Delta_{HR_F} - \Delta_{HR_M}|$ in Table 9 we report the difference between the drop (or gain) experienced by both genders. It can be seen that this value is the lowest for PerBlur with predicted ratings. The exception is ML1M where the value for BlurMe is lowest. In this case, PerBlur with predicting ratings outperforms BlurMe for both genders, so the gap is less worrisome. In general, PerBlur appears somewhat better in preventing obfuscation from widening the gender performance gap. We interpret this finding as reflecting the benefit of attempting to avoid obfuscating with "noise", but instead keep obfuscation as close as possible to what users might have done themselves.

Finally we note that here again we see that PerBlur with predicted ratings is superior to PerBlur with average ratings. In the remainder of the paper, we examine PerBlur using predicted ratings.

8. Achieving diverse results

In this section, we look at the impact of obfuscation on diversity. We first need an overview of the different variants of obfuscation we will investigate. We start by looking at the conditions for which we report Top-N recommendation performance. For this, the relevant performance levels were already reported in Table 8. Then, we will look at obfuscation with removal. For this purpose the Top-N recommendation performance is provided in Table 10. This table includes results for both random and greedy removal. We include this table here because obfuscation with removal is not discussed in detail in Section 6.2 due to the fact that our experiments showed that it did not have a consistent influence on recommendation performance or fairness. However, we study removal now because of its potential for enhancing diversity. Recall that the difference between the two removal strategies is that the random removal strategy removes items randomly from individual user profile and the greedy removal strategy removes items in the order of their gender indicativeness (in L_m and L_f) from individual user profile. In Table 10, we see that greedy removal and random removal are largely comparable. In our analysis of diversity we will argue that the choice should be made by taking diversity, and not just recommendation accuracy, into consideration.

Now that we have a complete view of recommendation performance for all the relevant variants of obfuscation, we dive into the impact of PerBlur on diversity. Remember that we study diversity by looking at the ability of PerBlur data to steer

Table 10
Ranking prediction results measured in terms of Top10.nDCG and HR@10 using BPRMF on Original, BlurMe, and PerBlur with removal Data (personalization with Top-50 indicative items and removal strategy). The scores report the average over five repetitions of the evaluation. The standard deviation of Top10.nDCG and HR@10 is around 0.001 on ML1M and Flixster data sets, and 0.003 on LastFM data.

BPRMF		*Obfuscation strategies*		*nDCG*	*HR@10*
Data sets		**Personalization**	**Removal**		
ML1M	*Original*	*None*	*None*	0.1632	0.1720
	PerBlur with Removal	*Personalized with Average Ratings*	*Random*	0.1591	0.1682
			Greedy	0.1545	0.1615
		Personalized with Predicted Ratings	*Random*	**0.1593**	**0.1678**
			Greedy	0.1534	0.1606
Flixster	*Original*	*None*	*None*	0.1159	0.0610
	PerBlur with Removal	*Personalized with Average Ratings*	*Random*	0.1092	0.0600
			Greedy	0.1056	0.0584
		Personalized with Predicted Ratings	*Random*	**0.1104**	**0.0607**
			Greedy	0.1073	0.0590
LastFM	*Original*	*None*	*None*	0.0882	0.0622
	PerBlur with Removal	*Personalized with Interactions*	*Random*	0.0764	0.0592
			Greedy	**0.0833**	**0.0576**

Table 11
Diversity for Standard PerBlur: The proportion of correctly recommended items that are stereotypical for gender (female and male) using BPRMF. Three different cutoff levels (10, 20, 50) are used to define gender-stereotypical items. Original Data and Standard PerBlur Data (personalization with Top-50 indicative items). The scores report the average over five repetitions of the evaluation. The standard deviation is around: 0.0002 on ML1M data, 0.0005 on Flixster data, and 0.0005 on LastFM data.

BPRMF		*Obfuscation strategies*		*Stereotypical gender items*					
Data sets		**Level**	**Personalization**	top10F	top10M	top20F	top20M	top50F	top50M
ML1M	*Original*	*0%*	*None*	0.0021	0.0044	0.0040	0.0068	0.0083	0.0127
	Standard PerBlur	*5%*	*Personalized with Predicted Ratings*	0.0020	0.0046	0.0036	0.0070	0.0077	0.0129
Flixster	*Original*	*0%*	*None*	0.0056	0.0090	0.0114	0.0150	0.0244	0.0266
	Standard PerBlur	*10%*	*Personalized with Predicted Ratings*	0.0038	0.0086	0.0083	0.0142	0.0197	0.0251
LastFM	*Original*	*0%*	*None*	0.001	0.0000	0.001	0.0000	0.0026	0.0002
	Standard PerBlur	*5%*	*Personalized with Interactions*	0.000	0.0000	0.000	0.0000	0.0003	0.0000

recommender systems away from providing gender-stereotypical recommendations. Recall also that we define a gender-stereotypical recommendation as an item that is highly typical for a particular gender. Our assumption is that users will appreciate a less stereotyped recommender, i.e., that women will appreciate when recommendations do not focus on stereotypical female items such as ‘chick flicks’. We are not looking to eliminate gender stereotypical items from the recommendation lists, but rather to control them.

For our analysis, we assume gender-stereotypical items to be items that are specific to a user’s gender. We make use of the lists of gender-indicative items, L_f and L_m, that we use for the gender obfuscation algorithms. Because these lists were derived before the training/test split, test items of users occur in these lists. Refer back to Fig. 3 to understand the way in which the training and test set are ensured to be disjoint. We test three different cutoffs for defining a list of gender-specific items: top10, top20, and top50 most specific items. Recall that PerBlur removes gender specific items from the training data. Note, however, that this does not impact the test data, which is a constant item set over all data sets tested (Fig. 3).

In Tables 11 and 12, we report the proportion of correctly recommended test-items that are gender-stereotypical. For PerBlur, these tables report the PerBlur variant that uses predicted ratings so as not to crowd the table. We choose this variant because it generally achieves better performance. The proportions in these tables are small because only a small number of top10, top20 or top50 items are in the ground truth. However, the relative difference between these proportions demonstrates the effect of PerBlur.

Table 11 corresponds to the recommender performance in Table 8. In Table 11, we see that PerBlur seems to lower the Top-N gender-stereotypical items that are recommended to both male and female users with respect to the original data.

Table 12 corresponds to the recommender performance in Table 10. Note that the performance on the original data is different between Tables 8 and 10. This difference arises because the conditions in these tables were run as two separate condition sets, which means that their candidate sets are not comparable, as was described in Section 6.1. In Table 12, we see that PerBlur with greedy

Table 12
Diversity for PerBlur with removal: The proportion of correctly recommended items that are stereotypical for gender (female and male) using BPRMF. Three different cutoff levels (10, 20, 50) are used to define gender-stereotypical items. Original Data and PerBlur Data (personalization with Top-50 indicative items using predicted ratings, and with **greedy removal**). The scores report the average over five repetitions of the evaluation. The standard deviation is around: 0.0002 on ML1M data, 0.0005 on Flixster data, and 0.0005 on LastFM data.

BPRMF		*Obfuscation strategies*		*Stereotypical gender items*					
Data sets		**Personalization**	**Removal**	top10F	top10M	top20F	top20M	top50F	top50M
ML1M	*Original*	*None*	*None*	0.0020	0.0045	0.0038	0.0069	0.0082	0.0128
	PerBlur with removal	*Personalized with Predicted Ratings*	*Random*	0.0017	0.0045	0.0033	0.0070	0.0075	0.0127
			Greedy	**0.0003**	**0.0005**	**0.0014**	**0.0020**	**0.0051**	**0.0073**
Flixster	*Original*	*None*	*None*	0.0058	0.0084	0.0115	0.0147	0.0225	0.0255
	PerBlur with removal	*Personalized with Predicted Ratings*	*Random*	0.0048	0.0087	0.0097	0.0149	0.0219	0.0265
			Greedy	**0.0006**	**0.0018**	**0.0035**	**0.0068**	**0.0149**	**0.0169**
LastFM	*Original*	*None*	*None*	0.0013	0.0010	0.0013	0.0010	0.0026	0.0027
	PerBlur with removal	*Personalized with Interactions*	*Random*	0.0013	0.0010	0.0013	0.0010	0.0013	0.0020
			Greedy	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0008**	**0.0012**

removal is highly effective in lowering the proportion of Top-N gender-stereotypical items. Random removal has no apparent impact on diversity. This effect can be attributed to the fact that greedy removal uses information about gender specificity and can guide the recommender away from gender-typical items. In sum, these results demonstrate the potential of using obfuscation to improve diversity at the same time as it is protecting users' privacy.

9. Conclusion and outlook

In this section, we summarize the main findings of our paper and also provide an outlook onto future working.

9.1. Summary

We have introduced PerBlur, a new gender obfuscation approach for recommender system data. PerBlur extends the state of the art with its use of personalization and also greedy item removal.

Main finding. The main contribution of the paper is a demonstration that PerBlur can maintain recommender system performance, and in some cases improve it, while also blocking the inference of gender information.

We have also shown that BlurMe, an approach that does not use personalization, is effective when applied to the entire user–item matrix, which was not previously demonstrated in the literature. The picture that emerges is that PerBlur shows advantages over BlurMe, but that BlurMe is also more effective than is expected.

User-oriented paradigm. The PerBlur approach was formulated within a user-oriented paradigm for user profile privacy. This paradigm requires approaches to be *understandable* to users, as well as remaining *unobtrusive* so that they do not get in the user's way. These are two desirable characteristics informed the design of PerBlur. The paradigm also requires that privacy look at *usefulness* as going beyond accuracy to encompass also fairness and diversity aspects of recommender system data protection.

Fairness and diversity. Our experiments have shown that obfuscation interacts with fairness and diversity. When using obfuscation it is important to check that different user groups are impacted in the same way. In our experiments, we saw that PerBlur appears to have an advantage over BlurMe in controlling the difference of the impact. We also showed that PerBlur has the potential to improve recommendation diversity by reducing the percentage of gender-stereotypical items that are recommended.

Evaluation methodology. We have pointed out that fairness of experimental analysis when testing obfuscated recommender system data is non-trivial. To address this challenge, we have proposed an evaluation procedure for obfuscated recommender system data, and carried out our experiments using this procedure.

9.2. Future work

The user-oriented paradigm for privacy protection offers a framework in which future work can formulate new approaches to protecting user data. The formulation of PerBlur itself is independent of the specific nature of the data and the attribute being protected. In the future, PerBlur could be applied also to different demographic attributes such as age, occupation, ethnicity, political orientation. Here, we elaborate further on the insights of this paper that are important for future work.

Obfuscation that promotes fairness and diversity. Work focusing on obfuscation of other attributes has been carried out, for example by Beigi et al. (2020), Chen et al. (2014), but not all of these approaches focus on data set obfuscation, and none of them investigate the potential benefits for fairness and diversity. In this respect, our paper opens an important new vista for future work.

From obfuscation to data synthesis. We also mention that PerBlur can be considered to be between obfuscation and data synthesis. Because it imputes items, PerBlur is effectively synthesizing a profile extension for each user. If obfuscation can take the data far enough away from the original user profile, but still keep it faithful to underlying distributions, it could become an important tool in creating data sets that can be released for the research community to use in the development of new algorithms.

Moving towards more sophisticated threat models. Our work is based on the threat model that is defined in Section 1.2. This threat model can be refined in the future to match more closely with real-world threats, in particular data breaches. A limitation of our current work is that we test gender classifiers with only one data set. A more sophisticated threat model would assume that different sources and different amounts of labeled data may be at the disposal of the attacker.

We close by summarizing the main insights that we have found important for guiding future work on data obfuscation.

Obfuscation is relatively easy. First, obfuscation is a simpler task than one might think. A simple approach, fully understandable to users, works well. An approximate setting of an operating point gives a practically useful approach. Future research should not blindly assume that the problem of gender obfuscation requires iterative optimization approaches. Such approaches are not only difficult for the user to understand, but they are computationally heavy and require recomputation as users continue to rate and interact with items.

Obfuscation need not add noise. Second, we should not assume that obfuscation must introduce noise. In this paper we have shown, that if we keep obfuscation close to user preferences it has the potential to be unobtrusive for the user and also allows us to maintain or even improve upon the performance of the original data.

Obfuscation should go beyond accuracy. Third, maintaining recommender system accuracy should not be the sole goal of obfuscation. Instead, fairness must also be maintained. We have seen that obfuscation also opens up an interesting opportunity to improve recommender system diversity.

Acknowledgments

We would like to thank Alejandro Bellogin and Babak Loni for their useful suggestions on building the top-N evaluation setup. Also, we would like to thank Christopher Strucks for the collaboration during his master thesis, which planted the seed for this work.

References

Abdollahpouri, H., Adomavicius, G., Burke, R., Guy, I., Jannach, D., Kamishima, T., et al. (2019). Beyond personalization: Research directions in multistakeholder recommendation. arXiv preprint arXiv:1905.01986.

Anelli, V. W., Deldjoo, Y., Di Noia, T., Ferrara, A., & Narducci, F. (2021). Federank: User controlled feedback with federated recommender systems. In D. Hiemstra, M.-F. Moens, J. Mothe, R. Perego, M. Potthast, & F. Sebastiani (Eds.), *Advances in information retrieval* (pp. 32–47).

Ardagna, C. A., Cremonini, M., Damiani, E., Di Vimercati, S. D. C., & Samarati, P. (2007). Location privacy protection through obfuscation-based techniques. In *IFIP annual conference on data and applications security and privacy* (pp. 47–60).

Badsha, S., Yi, X., & Khalil, I. (2016). A practical privacy-preserving recommender system. *Data Science and Engineering*, *1*(3), 161–177.

Badsha, S., Yi, X., Khalil, I., & Bertino, E. (2017). Privacy preserving user-based recommender system. *IEEE 37th international conference on distributed computing systems* (pp. 1074–1083).

Beigi, G., Mosallanezhad, A., Guo, R., Alvari, H., Nou, A., & Liu, H. (2020). Privacy-aware recommendation with private-attribute protection using adversarial learning. In *Proceedings of the 13th ACM international conference on web search and data mining* (pp. 34–42).

Bellogin, A., Castells, P., & Cantador, I. (2011). Precision-oriented evaluation of recommender systems: An algorithmic comparison. In *Proceedings of the 5th ACM conference on recommender systems* (pp. 333–336).

Berkovsky, S., Kuflik, T., & Ricci, F. (2012). The impact of data obfuscation on the accuracy of collaborative filtering. *Expert Systems with Applications*, *39*(5), 5033–5042.

Bertin-Mahieux, T., Ellis, D. P., Whitman, B., & Lamere, P. (2011). The million song dataset. In *Proceedings of the 12th international society for music information retrieval conference*.

Bertsimas, D., Pawlowski, C., & Zhuo, Y. D. (2017). From predictive methods to missing data imputation: an optimization approach. *Journal of Machine Learning Research*, *18*(1), 7133–7171.

Bhagat, S., Weinsberg, U., Ioannidis, S., & Taft, N. (2014). Recommending with an agenda: active learning of private attributes using matrix factorization. In *Proceedings of the 8th ACM conference on recommender systems* (pp. 65–72).

Bi, B., Shokouhi, M., Kosinski, M., & Graepel, T. (2013). Inferring the demographics of search users: Social data meets search queries. In *Proceedings of the 22nd ACM international conference on world wide web* (pp. 131–140).

Brunton, F., & Nissenbaum, H. (2015). *Obfuscation: A user's guide for privacy and protest*. MIT Press.

Burke, R. (2017). Multisided fairness for recommendation. In *FATREC 2017 workshop on fairness, accountability, and transparency in recommender systems, in conjunction with the 11th ACM conference on recommender systems*.

Burke, R., O'Mahony, M. P., & Hurley, N. J. (2015). Robust collaborative recommendation. In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender systems handbook* (pp. 961–995).

Burke, R., Sonboli, N., & Ordonez-Gauger, A. (2018). Balanced neighborhoods for multi-sided fairness in recommendation. In S. A. Friedler, & C. Wilson (Eds.), *Proceedings of machine learning research*: *vol. 81*, *Proceedings of the 1st conference on fairness, accountability and transparency* (pp. 202–214).

Calandrino, J. A., Kilzer, A., Narayanan, A., Felten, E. W., & Shmatikov, V. (2011). "You might also like:" privacy risks of collaborative filtering. In *IEEE Symposium on security and privacy* (pp. 231–246).

Casino, F., Domingo-Ferrer, J., Patsakis, C., Puig, D., & Solanas, A. (2015). A k-anonymous approach to privacy preserving collaborative filtering. *Journal of Computer and System Sciences*, *81*(6), 1000–1011.

Castells, P., Hurley, N. J., & Vargas, S. (2015). Novelty and diversity in recommender systems. In *Recommender systems handbook* (pp. 881–918).

Chen, T., Boreli, R., Kaafar, M.-A., & Friedman, A. (2014). On the effectiveness of obfuscation techniques in online social networks. In *International symposium on privacy enhancing technologies symposium* (pp. 42–62).

Cremonesi, P., Koren, Y., & Turrin, R. (2010). Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the 4th ACM conference on recommender systems* (pp. 39–46).

Deldjoo, Y., Di Noia, T., & Merra, F. A. (2021). A survey on adversarial recommender systems: From attack/defense strategies to generative adversarial networks. *ACM Computing Surveys*, *54*(2).

Dwork, C. (2008). Differential privacy: A survey of results. In M. Agrawal, D. Du, Z. Duan, & A. Li (Eds.), *Theory and applications of models of computation* (pp. 1–19).

Ekstrand, M. D. (2020). LensKit for Python: Next-generation software for recommender systems experiments. In *Proceedings of the 29th ACM international conference on information & knowledge management* (pp. 2999–3006).

Ekstrand, M. D., Joshaghani, R., & Mehrpouyan, H. (2018). Privacy for all: Ensuring fair and equitable privacy protections. In S. A. Friedler, & C. Wilson (Eds.), *Proceedings of machine learning research*: *vol. 81*, *Proceedings of the 1st conference on fairness, accountability and transparency* (pp. 35–47).

Ekstrand, M. D., & Kluver, D. (2021). Exploring author gender in book rating and recommendation. *User Modeling and User-Adapted Interaction*, 1–44.

Ekstrand, M. D., Tian, M., Azpiazu, I. M., Ekstrand, J. D., Anuyah, O., McNeill, D., et al. (2018). All the cool kids, how do they fit in?: Popularity and demographic biases in recommender evaluation and effectiveness. In *Proceedings of machine learning research*: *vol. 81*, *Proceeding FAT** (pp. 172–186).

Ekstrand, M. D., Tian, M., Kazi, M. R. I., Mehrpouyan, H., & Kluver, D. (2018). Exploring author gender in book rating and recommendation. *Proceedings of the 12th ACM conference on recommender systems* (pp. 242–250).

Epps-Darling, A., Bouyer, R. T., & Cramer, H. (2020). Artist gender representation in music streaming. In *Proceedings of the 21st international society for music information retrieval conference*.

Feng, T., Guo, Y., & Chen, Y. (2015a). Can user gender and recommendation performance be preserved simultaneously? In *IEEE International conference on computing, networking and communications* (pp. 227–231).

Feng, T., Guo, Y., & Chen, Y. (2015b). Can user privacy and recommendation performance be preserved simultaneously? *Computer Communications*, *68*, 17–24.

Feng, T., Guo, Y., Chen, Y., Tan, X., Xu, T., Shen, B., et al. (2014). Tags and titles of videos you watched tell your gender. In *IEEE international conference on communications* (pp. 1837–1842).

Ferraro, A., Serra, X., & Bauer, C. (2021). Break the loop: gender imbalance in music recommenders. In *Proceedings of the international conference on human information interaction and retrieval* (pp. 249–254).

Friedler, S. A., Scheidegger, C., & Venkatasubramanian, S. (2021). The (im)possibility of fairness: Different value systems require different mechanisms for fair decision making. *Communication of the ACM*, *64*(4), 136–143.

Friedman, A., Berkovsky, S., & Kaafar, M. A. (2016). A differential privacy framework for matrix factorization recommender systems. *User Modeling and User-Adapted Interaction*, *26*(5), 425–458.

Friedman, A., Knijnenburg, B. P., Vanhecke, K., Martens, L., & Berkovsky, S. (2015). Privacy aspects of recommender systems. In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender systems handbook* (pp. 649–688).

Gong, N. Z., & Liu, B. (2018). Attribute inference attacks in online social networks. *ACM Transactions on Privacy and Security*, *21*(1).

Hansen, C., Mehrotra, R., Hansen, C., Brost, B., Maystre, L., & Lalmas, M. (2021). Shifting consumption towards diverse content on music streaming platforms. *Proceedings of the 14th ACM international conference on web search and data mining* (pp. 238–246).

Harper, F. M., & Konstan, J. A. (2015). The MovieLens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, *5*(4).

Helberger, N., Karppinen, K., & D'Acunto, L. (2018). Exposure diversity as a design principle for recommender systems. *Information, Communication & Society*, *21*(2), 191–207.

Hu, G., & Yang, Q. (2020). PrivNet: Safeguarding private attributes in transfer learning for recommendation. In *Findings of the association for computational linguistics: EMNLP* (pp. 4506–4516).

Hua, J., Xia, C., & Zhong, S. (2015). Differentially private matrix factorization. In *Proceedings of the 24th International conference on artificial intelligence* (pp. 1763–1770).

Jeckmans, A. J., Beye, M., Erkin, Z., Hartel, P., Lagendijk, R. L., & Tang, Q. (2013). Privacy in recommender systems. In *Social media retrieval* (pp. 263–281).

Jia, J., Wang, B., Zhang, L., & Gong, N. Z. (2017). AttriInfer: inferring user attributes in online social networks using Markov random fields. In *ACM international world wide web conferences* (pp. 1561–1569).

Kaminskas, M., & Bridge, D. (2016). Diversity, serendipity, novelty, and coverage: A survey and empirical analysis of beyond-accuracy objectives in recommender systems. *ACM Transactions on Interactive Intelligent Systems*, *7*(1).

Kandappu, T., Friedman, A., Boreli, R., & Sivaraman, V. (2014). PRivacyCanary: Privacy-aware recommenders with adaptive input obfuscation. In *22nd international symposium on modelling, analysis simulation of computer and telecommunication systems* (pp. 453–462).

Kosinski, M., Stillwell, D., & Graepel, T. (2013). Private traits and attributes are predictable from digital records of human behavior. *Proceedings of the National Academy of Sciences*, *110*(15), 5802–5805.

Kunaver, M., & Požrl, T. (2017). Diversity in recommender systems – A survey. *Knowledge-Based Systems*, *123*, 154–162.

Lakshminarayan, K., Harp, S. A., & Samad, T. (1999). Imputation of missing data in industrial databases. *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies*, *11*(3), 259–275.

Li, Y., Vishwamitra, N., Knijnenburg, B. P., Hu, H., & Caine, K. (2017). Effectiveness and users' experience of obfuscation as a privacy-enhancing technology for sharing photos. *Proceeding of the ACM Human-Computing Interaction*, *1*.

Li, Q., Zheng, X., & Wu, X. (2017). Collaborative autoencoder for recommender systems. ArXiv E-Prints.

Mansoury, M., Abdollahpouri, H., Pechenizkiy, M., Mobasher, B., & Burke, R. (2020). FairMatch: A graph-based approach for improving aggregate diversity in recommender systems. In *Proceedings of the 28th ACM conference on user modeling, adaptation and personalization*. (pp. 154–162).

Mansoury, M., Abdollahpouri, H., Smith, J., Dehpanah, A., Pechenizkiy, M., & Mobasher, B. (2020). Investigating potential factors associated with gender discrimination in collaborative recommender systems. *Proceedings of the 13th FLAIRS conference*.

McSherry, F., & Mironov, I. (2009). Differentially private recommender systems: Building privacy into the Netflix prize contenders. In *Proceedings of the 15th international conference on knowledge discovery and data mining* (pp. 627–636).

Mislove, A., Viswanath, B., Gummadi, K. P., & Druschel, P. (2010). You are who you know: inferring user profiles in online social networks. In *Proceedings of the 3rd ACM international conference on web search and data mining* (pp. 251–260).

Mobasher, B., Burke, R., Bhaumik, R., & Williams, C. (2007). Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transaction Internet Technology*, *7*(4), 23–es.

Narayanan, A., & Shmatikov, V. (2008). Robust de-anonymization of large sparse datasets. In *IEEE symposium on security and privacy* (pp. 111–125).

Nikolaenko, V., Ioannidis, S., Weinsberg, U., Joye, M., Taft, N., & Boneh, D. (2013). Privacy-preserving matrix factorization. In *Proceedings of the international conference on computer & communications security* (pp. 801–812).

Oliveira, R. S., Nóbrega, C., Marinho, L. B., & Andrade, N. (2017). A multiobjective music recommendation approach for aspect-based diversification. In *Proceedings of the 18th international society for music information retrieval conference* (pp. 414–420).

Parameswara, R., & Blough, D. M. (2007). Privacy preserving collaborative filtering using data obfuscation. In *International conference on granular computing* (p. 380).

Parra-Arnau, J., Rebollo-Monedero, D., & Forné, J. (2014). Optimal forgery and suppression of ratings for privacy enhancement in recommendation systems. *Entropy*, *16*(3), 1586–1631.

Pilászy, I., Zibriczky, D., & Tikk, D. (2010). Fast als-based matrix factorization for explicit and implicit feedback datasets. In *Proceedings of the 4th ACM conference on recommender systems* (pp. 71–78).

Polat, H., & Du, W. (2003). Privacy-preserving collaborative filtering using randomized perturbation techniques. In *3rd IEEE international conference on data mining* (pp. 625–628).

Qiang, Y. (2019). Federated recommendation systems. In *IEEE international conference on big data* (p. 1).

Reddy, S., & Knight, K. (2016). Obfuscating gender in social media writing. In *Proceedings of the 2016 EMNLP workshop on NLP and computational social science* (pp. 17–26).

Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th conference on uncertainty in artificial intelligence* (pp. 452–461).

Resheff, Y. S., Elazar, Y., Shahar, M., & Shalom, O. S. (2018). Privacy and fairness in recommender systems via adversarial training of user representations. arXiv preprint arXiv:1807.03521.

Salamatian, S., Zhang, A., d. P. Calmon, F., Bhamidipati, S., Fawaz, N., Kveton, B., et al. (2013). How to hide the elephant- or the donkey- in the room: Practical privacy against statistical inference for large data. In *Global conference on signal and information processing* (pp. 269–272).

Salamatian, S., Zhang, A., du Pin Calmon, F., Bhamidipati, S., Fawaz, N., Kveton, B., et al. (2015). Managing your private and public data: Bringing down inference attacks against your privacy. *IEEE Journal of Selected Topics in Signal Processing*, *9*(7), 1240–1255.

Salter, C., Saydjari, O. S., Schneier, B., & Wallner, J. (1998) Toward a secure system engineering methodology. In *Proceedings of the 1998 workshop on new security paradigms* (pp. 2–10).

Shakespeare, D., Porcaro, L., Gómez, E., & Castillo, C. (2020). Exploring artist gender bias in music recommendation. In *workshop on the impact of recommender systems (ImpactRS) in conjunction with the 14th ACM conference on recommender systems*.

Slokom, M., Larson, M., & Hanjalic, A. (2019). Data masking for recommender systems: prediction performance and rating hiding. In *Late breaking results, in conjunction with the 13th ACM conference on recommender systems*.

Strucks, C., Slokom, M., & Larson, M. BlurM(or)e: revisiting gender obfuscation in the user-item matrix. In *Recommendation in multi-stakeholder environments (RMSE), in conjunction with the 13th ACM conference on recommender systems*.

Su, X., Khoshgoftaar, T. M., & Greiner, R. (2008). Imputed neighborhood based collaborative filtering. *1*, In *2008 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology* (pp. 633–639).

Vargas, S., & Castells, P. (2011). Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the 5th ACM conference on recommender systems* (pp. 109–116).

Weinsberg, U., Bhagat, S., Ioannidis, S., & Taft, N. (2012). BlurMe: Inferring and obfuscating user gender based on ratings. In *Proceedings of the 6th ACM conference on recommender systems* (pp. 195–202).

Wu, Y., DuBois, C., Zheng, A. X., & Ester, M. (2016). Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the 9th ACM international conference on web search and data mining* (pp. 153–162).

Yang, D., Qu, B., & Cudré-Mauroux, P. (2019). Privacy-preserving social media data publishing for personalized ranking-based recommendation. *IEEE Transactions on Knowledge and Data Engineering*, *31*(3), 507–520.

Yao, S., & Huang, B. (2017). Beyond parity: Fairness objectives for collaborative filtering. In *Proceedings of the 31st international conference on neural information processing systems* (pp. 2925–2934).

Yuan, X., Han, L., Qian, S., Xu, G., & Yan, H. (2019). Singular value decomposition based recommendation using imputed data. *Knowledge-Based Systems*, *163*, 485–494.

Zafarani, R., & Liu, H. (2009). Social computing data repository at ASU.