2021

# Tools for responsible decision-making in machine learning

BOSTON UNIVERSITY

GRADUATE SCHOOL OF ARTS AND SCIENCES

Dissertation

# TOOLS FOR RESPONSIBLE DECISION-MAKING IN MACHINE LEARNING

by

**BASHIR RASTEGARPANAH**

B.Sc., University of Isfahan, 2010
M.Sc., University of Bonn, 2014

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2021

## Approved by

First Reader

Mark Crovella, PhD
Professor of Computer Science, Boston University

Second Reader

Krishna P. Gummadi, PhD
Professor of Computer Science, Saarland University

# Acknowledgments

I would like to thank the following people, without whom I would not have been able to complete this dissertation.

First of all, I would like to thank my advisor, Mark Crovella, for his consistent support and guidance throughout my PhD years. It was a great pleasure having the opportunity to learn from Mark's extensive knowledge, outstanding research experience, and insatiable curiosity. I would also like to thank my co-advisor, Krishna Gummadi, for introducing me to the field of responsible machine learning, and for his critical role in directing this work.

I am also very grateful to Evimaria Terzi and Adam Smith, not only for being part of my dissertation committee, but also for teaching me topics in algorithmic data mining and privacy in machine learning, which I later used in my research.

Many thanks to all the awesome friends I have made during these years, especially the former PhD students who taught me those ins and outs of doing a PhD that I had no other way to learn. In particular, I would like to thank Giovanni Comarela, Natali Ruchansky, Larissa Spinelli, Nabeel Akhtar, Harry Mavroforakis, Behzad Golshan, Zhenyu Liao, Mehrnoosh Sameki, Sofia Nikolakaki, and Harshal Chaudhari.

Special thanks to the staff at Boston University CS department, and the Max Planck Institute SWS for their effort in providing me with an ideal work environment during my PhD studies. Furthermore, I would like to thank the National Science Foundation and the European Research Council for their financial support.

Finally, I am extremely grateful to my parents, Malihe and Amir, for their unconditional love and support, and the sacrifices they have made for my success.

# TOOLS FOR RESPONSIBLE DECISION-MAKING IN MACHINE LEARNING

BASHIR RASTEGARPANAH

Boston University, Graduate School of Arts and Sciences, 2021

Major Professor: Mark Crovella, PhD
Professor of Computer Science

ABSTRACT

Machine learning algorithms are increasingly used by decision making systems that affect individual lives in a wide variety of ways. Consequently, in recent years concerns have been raised about the social and ethical implications of using such algorithms. Particular concerns include issues surrounding privacy, fairness, and transparency in decision systems. This dissertation introduces new tools and measures for improving the social desirability of data-driven decision systems, and consists of two main parts.

The first part provides a useful tool for an important class of decision making algorithms: collaborative filtering in recommender systems. In particular, it introduces the idea of improving socially relevant properties of a recommender system by augmenting the input with additional training data, an approach which is inspired by prior work on data poisoning attacks and adapts them to generate 'antidote data' for social good. We provide an algorithmic framework for this strategy and show that it can efficiently improve the polarization and fairness metrics of factorization-based recommender systems.

In the second part, we focus on fairness notions that incorporate data inputs used by decision systems. In particular, we draw attention to 'data minimization', an existing principle in data protection regulations that restricts a system to use the minimal information that is necessary for performing the task at hand. First, we propose an operationalization for this principle that is based on classification accuracy, and we show how a natural dependence of accuracy on data inputs can be expressed as a trade-off between fair-inputs and fair-outputs. Next, we address the problem of auditing black- box prediction models for data minimization compliance. For this problem, we suggest a metric for data minimization that is based on model instability under simple imputations, and we extend its applicability from a finite sample model to a distributional setting by introducing a probabilistic data minimization guarantee. Finally, assuming limited system queries, we formulate the problem of allocating a query budget to simple imputations for investigating model instability as a multi-armed bandit framework, for which we design efficient exploration strategies.

# Contents

# List of Tables

# List of Figures

# List of Abbreviations

| | | |
|---|---|---|
| DT | ............. | Decision Tree |
| GA | ............. | Gradient Ascent |
| GD | ............. | Gradient Descent |
| GDPR | ............. | General Data Protection Regulation |
| PM | ............. | Probability Matching |
| TS | ............. | Thompson Sampling |
| TTTS | ............. | Top-Two Thompson Sampling |

# Chapter 1

# Introduction

## 1.1 Responsible Machine Learning

Over the past few decades, machine learning has been transformed from a mostly academic field to a technology that is integrated into all kinds of services and devices used by the public. Nowadays, machine learning algorithms largely shape the choices we make in our daily lives ranging from what news we read and which movies we watch, to whose products we buy. They are also used to make decisions such as whether an applicant should be offered a loan, or a defendant should be detained while awaiting trial. The increasing use of such data-driven decision making systems in modern society has raised concerns about their social and ethical implications, particularly about issues surrounding privacy, fairness, and transparency.

These concerns are amplified by a series of recent empirical studies that show how algorithmic decision systems are prone to unfair treatment of their users. For example, Larson et al. (2016) analyzed an algorithmic tool used by U.S. courts to assess the likelihood of a defendant becoming a recidivist, and their study showed that black defendants were twice as likely as whites to be labeled as high-risk but do not actually re-offend. (Examples from other domains are (Buolamwini and Gebru, 2018), (Sweeney, 2013).)

These findings have raised public awareness about the need for responsible decision-making in machine learning, which is responded by regulatory authorities in different countries (House, 2016; Goddard, 2017). Such regulations not only aim to combat discrimination against some individuals or groups of users, but also highlight other user rights including privacy protection or the right to ask for explanations of system decisions.

Despite the efforts made by policy makers to address the above social concerns, there is often a lack of operational interpretation of social responsibility concepts and regulations for particular algorithmic decision systems. This in turn has created an active research area focusing on developing notions to define, measures to quantify, and methods to achieve certain socially desirable properties in decision systems. From the perspective of system designers, taking the social responsibility dimension into account translates into setting new objectives under which the system performance is evaluated. In other words, traditional learning objectives such as the expected loss over a population are no longer the only performance metrics to optimize for.

Recent years have witnessed a fast-growing number publications in the area of responsible machine learning. However, as stated by Chouldechova and Roth (2020) *"Our understanding of the fundamental questions related to fairness and machine learning remain in its infancy."* In particular, much work remains to be done in formalizing social and ethical objectives in different circumstances, developing efficient mechanisms for both achieving and auditing social objectives, and understanding the inherent trade-offs between different objectives. This thesis contributes in further developing all the above-mentioned directions. In the following we discuss the specific contributions of this thesis in details.

## 1.2 Contributions of This Thesis

The first contribution of this thesis (Chapter 3) is introducing a new mechanism for improving social desirability of an important class of decision making algorithms: collaborative filtering in recommender systems. As these algorithms rely on user-provided data to learn models that are used to predict unknown user preferences, the recommendations made by such systems may carry undesired properties which are inherent in the observed data. Existing approaches to mitigate this effect mainly rely on in-processing techniques, i.e., they modify the system by by using loss functions that incorporate the fairness objective (Burke et al., 2018; Kamishima et al., 2012). As a result, they require the recommender system to be modified each time a different social objective is considered.

We introduce a new method for improving socially relevant properties of a recommender system. Our approach is based on augmenting the system inputs with additional training data, an approach which is inspired by prior work on data poisoning attacks and adapts them to generate antidote data for social good. We develop a generic algorithmic framework for this strategy and show that it can efficiently improve the polarization and fairness metrics of factorization-based recommender systems.

As a strategy for improving recommendations, the data augmentation approach has multiple advantages. Adding new input data may be easier than modifying existing data inputs, as when a system is already running. Additional data can be provided to the system by a third-party who does not need the ability to modify the system's existing input, nor the ability to modify the system's algorithms. Finally, unlike existing in-processing approaches that require changing the learning objective for each desired objective, our approach is applicable to a wide range of socially relevant properties of a system.

The next contributions of this thesis are focused on the social concerns regarding the data inputs used by decision systems, which are less studied to date. In particular, we draw attention to 'data minimization', an existing principle in data protection regulations such as the GDPR (article 5.1.c) (Goddard, 2017). This principle restricts a system to use the minimal information that is necessary for performing the task at hand. However, interpreting (i.e., operationalizing) this principle for particular prediction systems remains largely unclear.

In Chapter 4 we propose an operationalization for data minimization principle that is based on classification accuracy. We call this operationalization the *need-to-know* property. Furthermore, we propose another input property, *fair-privacy*, which requires the decision system use the same data inputs about all individuals when making decisions. We explore the interaction between these properties and fairness in the outputs (fair prediction accuracy), and show that for an *optimal* classifier these three properties are in general incompatible. We explain what common properties of data make them incompatible and provide an algorithm to verify if the trade-off between the three properties exists in a given dataset. Given that achieving optimal accuracy is a traditional goal of designing classifiers, this result pose a new challenge to the design of classifiers that aim at optimality: "how much one needs to compromise on optimality in order to simultaneously achieve fairness in the inputs and outputs of a classifier?"

Finally, Chapter 5 addresses the problem of auditing black-box prediction models for data minimization compliance. For this problem, we propose an operationalization that is based on model instability. Given the challenge of the black-box setting, our key idea is to check if each of the prediction model's input features are individually necessary, by simply imputing (i.e., assigning) them some constant value and measuring the extent to which the prediction model's outcomes would change. We introduce a metric for data minimization that is based on model instability under different simple imputations. we extend the applicability of this metric from a finite sample model to a distributional setting by introducing a probabilistic data minimization guarantee, which we derive using a Bayesian approach. Furthermore, we address the auditing problem under a constraint on the number of queries to the prediction system. We formulate the problem of allocating a query budget to feasible simple imputations for investigating model instability as a multi-armed bandit framework with probabilistic success metrics, for which we design efficient algorithms.

# Chapter 2

# Background

This chapter provides an overview of fairness in machine learning (related to Chapters 3 & 4), and the data minimization principle (related to Chapters 4 & 5) in the literature. The related work specific to each chapter is reviewed in the corresponding related work section of that Chapter.

## 2.1 Fairness in Machine Learning

In recent years several empirical studies have shown how data-driven decision algorithms may amount to discrimination against some users in different areas such as online advertisement (Sweeney, 2013) and criminal justice (Larson et al., 2016). More examples of such empirical studies are provided in survey papers (Romei and Ruggieri, 2014; Barocas and Selbst, 2016). These findings have raised awareness about the potential for social harm by the use of machine learning algorithms in life-affecting decision making scenarios and the importance of fair decision-making systems (Barocas and Selbst, 2016; Boyd and Crawford, 2012), which are highlighed by regulatory authorities as well (Munoz et al., 2016; Regulation, 2016).

One line of research on fair machine learning focuses on formulating fairness metrics. Numerous fairness notions have been proposed so far for machine learning tasks as varied as classification (Zafar et al., 2017a,b; Hardt et al., 2016; Zemel et al., 2013), regression (Berk et al., 2017), ranking (Biega et al., 2018; Singh and Joachims, 2018; Zehlike et al., 2017), and set selection (Celis et al., 2016). The proposed notions fall under two broad categories: those measuring unfairness at the level of *individual users* and those that measure unfairness at the level of *user groups* (Speicher et al., 2018; Dwork et al., 2012).

The group-level unfairness measures can be further sub-divided into those that prohibit the use of information related to a user's sensitive group membership when making predictions and those that require users belonging to different sensitive groups to receive, on average, *equal quality of service.* The quality of service received by user groups can in turn be measured either *conditioned or unconditioned* on the outcomes deserved by the users. That is, one might be concerned about the discrepancy between the rate of beneficial outcomes in different user groups regardless of the ground truth labels (disparate impact). On the other hand, when ground truth labels are available, considering the discrepancy in misclassification rates across different groups might be preferred (disparate mistreatment) (Zafar et al., 2017b).

Another active area in fair machine learning is developing techniques to improve the fairness of algorithmic systems. Fairness-enhancing techniques in general fall into three categories based on the stage of the machine learning pipeline that they are employed: (i) *pre-processing*,i.e., transform the training data to reduce the potential for unfair outcomes when using traditional learning models (Kamiran and Calders, 2012; Calmon et al., 2017), (ii) *in-processing*, i.e., change the learning objectives and models to ensure fair outcomes even using unmodified training data (Kamishima et al., 2011; Agarwal et al., 2018; Zafar et al., 2017b), and (iii) *post-processing*, i.e., modify potentially unfair outcomes from existing pre-trained learning models (Hardt et al., 2016; Corbett-Davies et al., 2017).

## 2.2 Data Minimization

Privacy in information systems is generally understood using two concepts: limitation theory and control theory (Tavani, 2007). Using those theories, several methods have been proposed to protect the privacy of the users in practice such as differential privacy (Dwork et al., 2014) k-anonymity (Aggarwal, 2005) and cryptography (Stinson, 2005). The proposed methods mostly assume a privacy adversary who is different from the party that collects and processes personal data. In many practical scenarios however, the decision system itself is considered as the privacy adversary. On the other hand, achieving *complete privacy* has been the goal of cryptography approaches such as secure multi-party computation (SMC), which have limited practicality due to constraints such as computational efficiency, and regulatory auditing purposes that require recording some user data (e.g., attributes such as race or gender for detecting discrimination).

Consequently, the alternative goal of acquiring minimum necessary data has become important as stated by regulations in different countries. In particular the EU General Data Protection Regulation (GDPR) (Regulation, 2016) defines the *data minimization* principle to control the extent to which personal data can be acquired and used by prediction models. This principle states that:

> *Personal data shall be adequate, relevant and limited to what is necessary for the purposes for which they are processed.* (article 5.1.c)

Only a few recent works have addressed the problem of operationalizing data minimization in prediction systems. Biega et al. (2020) propose definitions that are based on recommender systems' performance, and conduct an empirical study to check the original recommender performance can be preserved, while limiting the number of known user ratings. Galdon Clavell et al. (2020) interpret data minimization as limiting the use of sensitive personal data and do an experimental analysis that suggests data minimization should not be applied without consideration of other social concerns such as fairness.

Our focus here is on operationalizing data minimization, which is one of the users rights mentioned in the GDPR. The GDPR however consists of various principles for protecting individuals who interact with computational system, and efforts have been made for putting those principles into practice. Examples include proposals for operationalizing the right to explanation (Kaminski, 2019), the right to be forgotten (Koops, 2011; Ginart et al., 2019), the notion of singling out (Cohen and Nissim, 2020), and the right to withdraw consent (Politou et al., 2018; Utz et al., 2019).

# Chapter 3

# Antidote Data Framework

## 3.1 Introduction

Recommender systems are at the core of many online platforms that influence the choices we make in our daily lives ranging from what news we read (e.g., Facebook, Twitter) and whose products and services we buy (e.g., Amazon, Uber, Netflix) to whom we meet (e.g., OKCupid, Tinder). As users increasingly rely on recommender systems to make life-affecting choices, concerns are being raised about their inadvertent potential for social harm. Recently, studies have shown how recommender systems predicting user preferences might offer *unfair or unequal* quality of service to indvidual (or groups of) users (Beutel et al., 2017b; Burke et al., 2018) or lead to *societal polarization* by increasing the divergence between preferences of individual (or groups of) users (Dandekar et al., 2013).

Collaborative filtering recommender systems rely on user-provided data to learn models that are used to predict unknown user preferences. As a result, the recommendations made by such systems may carry undesired properties which are inherent in the observed data. A natural approach then is to consider transformations of input data that ameliorate those properties.

In this chapter we explore a new approach. Rather than transforming the system's exisiting input data, we investigate whether simply *augmenting the input with additional data* can improve the social desirability of the resulting recommendations. We explore this question by developing a generic framework that can be used to improve a variety of socially relevant properties of recommender systems. Our framework turns a technique that has previously been thought of as anti-social attacks on learning systems into a method with socially desirable outcomes.

As a strategy for improving recommendations, the data augmentation approach has multiple advantages. Adding new input data may be easier than modifying existing data inputs, as when a system is already running. Additional data can be provided to the system by a third-party who does not need the ability to modify the system's existing input, nor the ability to modify the system's algorithms. Further, the approach is applicable to a wide range of socially relevant properties of a system – essentially any property that can be expressed as a differentiable function of the systems inputs (ratings) and/or outputs (predictions).

The framework we develop starts from an existing matrix-factorization recommender system organized according to users and items, that has already been trained with some input (ratings) data. We consider the addition to the system of new users who provide ratings of existing items. The new users' ratings are chosen according to our framework, so as to improve a socially relevant property of the recommendations that are provided to the *original* users. We call the additional ratings provided 'antidote' data (by analogy to existing work studying data poisoning). While our presentation is organized in terms of adding new users to the system, it is entirely symmetric and can be equally applied to the addition of new items to the system.

In this thesis we instantiate the framework by proposing metrics that capture the polarization and unfairness of the system's recommendations. These metrics build on and extend previous proposals, and include measures of both individual and group unfairness. We show how to generate antidote data for these metrics, and we present a number of computational efficiencies that can be exploited. In the process we consider the relationship between improvements to socially-relevant measures and changes to overall system accuracy. Finally, we show that the small amounts of antidote data (typically on the order of 1% new users) can generate a dramatic improvement (on the order of 50%) in the polarization or the fairness of the system's recommendations.

## 3.2 Related Work

In this section, we first discuss how our measures of fairness and polarization for recommender systems relate to those discussed in prior works. Later, we describe how we leverage insights and methods explored in adversarial machine learning to cause social harm towards social good in recommender systems. For a general overview of fairness in machine learning see Section 2.1.

**Fairness in recommender systems.** The past years have witnessed a growing awareness about the potential for social harm by the use of machine learning algorithms in different areas (Barocas and Selbst, 2016; Boyd and Crawford, 2012). A detailed description of the related works in this area is provided in Chapter 2.

Compared to learning tasks such as classification and regression, few studies have explored fairness notions in the context of recommender systems. Recently, Burke et al. (2018) observed that recommender systems predicting user preferences over items would have to consider fairness from *two-sides* namely, from the perspective of *users* receiving the recommendations and from the perspective of *items* being recommended. Some of the early works by Kamishima et al. (2012, 2018); Kamishima and Akaho (2017) focused on notions of group-level fairness, where the learning model is modified to ensure that item recommendations are independent of users' features revealing sensitive group membership such as race and gender. More recently, Beutel et al. (2017b) and Yao and Huang (2017) have defined notions of group-level fairness in recommender systems based on the accuracy of predictions across different groupings of users or items.

Here, we not only build upon the group-level notions of fariness proposed by Beutel et al. (2017b) (by generalizing them to scenarios with more than two groups), but we also extend them to individual-level. We further note that our fairness notions can be applied either from the perspective of users or items.

**Mechanims for fair recommender systems.** Prior works have explored a number of approaches to incorporating fairness in learning models and recommender systems. These approaches can be broadly categorized into those that rely on (i) *pre-processing*, i.e., transform the training data to reduce the potential for unfair outcomes when using traditional learning models (Kamiran and Calders, 2012; Calmon et al., 2017), (ii) *in-processing*, i.e., change the learning objectives and models to ensure fair outcomes even using unmodified training data (Kamishima et al., 2011; Agarwal et al., 2018), and (iii) *post-processing*, i.e., modify potentially unfair outcomes from existing pre-trained learning models (Hardt et al., 2016; Corbett-Davies et al., 2017).

We explore a different approach to incorporating our fairness notions in recommender systems. Our approach is in contrast to existing approaches to fair recommendations that primarily rely on in-processing (Burke et al., 2018; Kamishima et al., 2012). Unlike in-processing approaches, our approach does not require us to modify the recommendation algorithm for each of our desired notions of fairness.

**Leveraging adversarial machine learning for social good.** Our approach relies on methods that have been traditionally used in adversarial learning literature to cause social harm (Huang et al., 2011). Our key insight is that we can retarget adversarial methods designed to “poison” training data and cause social harm to generate “antidote” training data for social good. Specifically, our antidote data generation methods are inspired by prior work on data poisoning attacks on factorization-based collaborative filtering (Li et al., 2016).

Most pre-processing approaches target learning new fair (latent and transformed) representations of original data. Recently, Beutel et al. (2017a) leveraged adversarial training procedures to remove information about sensitive group membership from the latent representations learned by a neural network. In contrast, our approach leaves the original training data untouched and instead adds new antidote data to achieve fairness objectives. As our evaluation results presented later in the chapter will show, by leaving the original training data unmodified, our approach also achieves good overall prediction accuracy (the traditional objective of recommender algorithms).

**Polarization.** Polarization refers to the degree to which opinions, views, and sentiments diverge within a population. Several prior works have raised and explored concerns that recommender systems might increase societal polarization by tailoring recommendations to individual user's preferences and trapping users in their own "personalized filter bubbles" (Pariser, 2011; Hannak et al., 2013). Dandekar et al. (2013) show how many traditional recommender algorithms used on Internet platforms can lead to polarization of user opinions in society.

We propose to measure the polarization of a recommender system as the extent to which predicted ratings for items vary (diverge) across users. Our polarization metric is consistent with those proposed in (Dandekar et al., 2013; Matakos et al., 2017). We show how our antidote data generation framework can be used to target reducing (or in certain scenarios, increasing) polarization in predicted ratings.

## 3.3 Optimal Antidote Data Problem

We start by presenting the system setup, notation, and problem definition. Assume $\mathbf{X} \in \mathbb{R}^{n \times d}$ is a partially observed rating matrix of $n$ users and $d$ items such that element $x_{ij}$ denotes the rating given by user $i$ to item $j$. Let $\Omega$ be the set of indices of known ratings in $\mathbf{X}$. Also $\Omega^i$ denotes the indices of known item ratings for user $i$, and $\Omega_j$ denotes the indices of known user ratings for item $j$.

For a matrix $\mathbf{A}$, $P_{\Omega}(\mathbf{A})$ is a matrix whose elements at $(i,j) \in \Omega$ are $a_{ij}$ and zero elsewhere. Similarly, for a vector $\mathbf{a}$, $P_{\Omega_j}(\mathbf{a})$ is a vector whose elements at $i \in \Omega_j$ are the corresponding elements of $\mathbf{a}$ and zero elsewhere. Throughout the chapter, we denote the column $j$ of $\mathbf{A}$ by the vector $\mathbf{a}_j$ and the row $i$ of $\mathbf{A}$ by the vector $\mathbf{a}^i$. All vectors are column vectors.

We assume a factorization based collaborative filtering algorithm is applied to estimate the unknown ratings in $\mathbf{X}$, i.e., for each user $i$ and item $j$ we find $\ell$-dimensional representations $\mathbf{u}_i$ and $\mathbf{v}_j$ such that $\ell << min(n,d)$ and the rating $x_{ij}$ is modeled by $x_{ij} \approx \mathbf{u}_i^{\intercal}\mathbf{v}_j$.

More specifically, we consider a factorization algorithm $\mathbf{\Theta}$ that finds factors $\mathbf{U} \in \mathbb{R}^{\ell \times n}$ and $\mathbf{V} \in \mathbb{R}^{\ell \times d}$ by solving the following optimization problem:

$$\underset{\mathbf{U},\mathbf{V}}{\operatorname{argmin}} \quad ||P_{\Omega}(\mathbf{X} - \mathbf{U}^{\intercal}\mathbf{V})||_F^2 + \lambda(||\mathbf{U}||_F^2 + ||\mathbf{V}||_F^2) \tag{3.1}$$

where columns of $\mathbf{U}$ are the user latent vectors, and columns of $\mathbf{V}$ are the item latent vectors. The first term in 3.1 denotes the estimation error over known elements of $\mathbf{X}$ and the second term is an $\ell_2$-norm regularizer added to avoid overfitting. The unknown ratings are then estimated by setting $\hat{\mathbf{X}} = \mathbf{U}^{\intercal}\mathbf{V}$.

We can think of our factorization algorithm as a function that maps a partially observed rating matrix $\mathbf{X}$ to matrices $\mathbf{U}$ and $\mathbf{V}$, and has additional parameters $\ell$ and $\lambda$, i.e, $\mathbf{\Theta}_{\ell,\lambda}(\mathbf{X}) = (\mathbf{U},\mathbf{V})$. We assume that the factorization rank and the regularizer parameter are set in a validation phase and remain fixed afterwards and we use $\mathbf{\Theta}(\mathbf{X})$ and $\mathbf{\Theta}_{\ell,\lambda}(\mathbf{X})$ interchangeably throughout the chapter.

We use $R$ to denote the socially relevant objective function that we seek to optimize by adding antidote data. $R$ is a function of estimated ratings $\hat{\mathbf{X}}$ and possibly (depending on the objective) other parameters such as original ratings, user labels, etc. For example, consider an objective that minimizes the difference of average es-

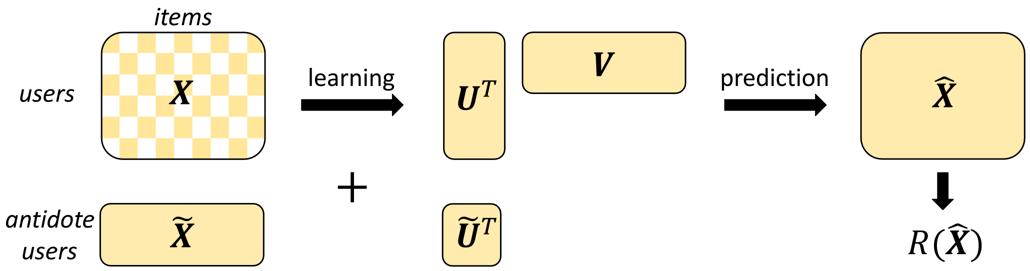


**Figure 3·1:** The effect of antidote data on a matrix factorization system. Initially the system learns factors $\mathbf{U}$ and $\mathbf{V}$ from a partially observed rating matrix $\mathbf{X}$. The latent factors are then used to find the estimated rating matrix $\hat{\mathbf{X}}$ which is an input to the socially relevant metric $R$. Adding antidote ratings $\tilde{\mathbf{X}}$ introduces the new user latent factor $\tilde{\mathbf{U}}$ and modifies the item latent factor $\mathbf{V}$, generating a new $\hat{\mathbf{X}}$ that improves $R(\hat{\mathbf{X}})$.

timation errors between two groups of users. In that case, $R$ is a function defined over $\mathbf{X}$, $\hat{\mathbf{X}}$, and another parameter that indicates the group membership of each user. The specific objective functions we study in this chapter are presented in Section 3.5. Now we can formally state the optimal antidote data problem:

**Problem 1** (Optimal Antidote Problem)**.** *Given a partially observed rating matrix* $\mathbf{X} \in \mathbb{R}^{n \times d}$*, a budget* $n' = \alpha n$*, a factorization algorithm* $\mathbf{\Theta}_{\ell,\lambda}$*, and an objective function* $R$*, find the antidote data* $\tilde{\mathbf{X}} \in \mathbb{R}^{n' \times d}$ *such that* $R$ *is optimized when* $\mathbf{\Theta}_{\ell,\lambda}$ *is applied jointly on* $\mathbf{X}$ *and* $\tilde{\mathbf{X}}$*.*

Note that we may want to either maximize or minimize $R$ depending on the objective. Also, although in our notation $\tilde{\mathbf{X}}$ corresponds to a set of artificial users, we can apply problem 1 to generate a set of artificial items by using the symmetry of the problem, i.e., by transposing $\mathbf{X}$.

Although some objective functions have additional parameters such as the original observed ratings ($\mathbf{X}$) or a list of group memberships (which we denote $K$), adding antidote data only affects the output of the factorization algorithm and hence the rating estimations $\hat{\mathbf{X}}$. Therefore, we denote the general objective function by $R(\hat{\mathbf{X}})$ instead of $R(\hat{\mathbf{X}}, \mathbf{X}, K)$ for notational convenience. Assuming our goal is to minimize some objective function $R$, we can rewrite problem 1 as:

$$\underset{\tilde{\mathbf{X}} \in \mathbb{M}}{\operatorname{argmin}} \quad R(\hat{\mathbf{X}}) \tag{3.2}$$

where $\mathbb{M} \subset \mathbb{R}^{n' \times d}$ is the set of feasible antidote data matrices.

Let $\mathbf{\Theta}(\mathbf{X}; \tilde{\mathbf{X}})$ denote the factorization algorithm when applied jointly on the original and the antidote data. In this case, the output consists of the item latent vectors forming the columns of factor $\mathbf{V} \in \mathbb{R}^{\ell \times d}$, and the user latent vectors which can be split into a matrix of original users latent vectors $\mathbf{U} \in \mathbb{R}^{\ell \times n}$, and a matrix of antidote users latent vectors $\tilde{\mathbf{U}} \in \mathbb{R}^{\ell \times n'}$; therefore, we have $\mathbf{\Theta}(\mathbf{X}; \tilde{\mathbf{X}}) = (\mathbf{U}, \tilde{\mathbf{U}}, \mathbf{V})$.

Furthermore, $\hat{\mathbf{X}}$ is a function of original users latent vectors and item latent vectors[1], i.e., $\hat{\mathbf{X}}(\mathbf{\Theta}(\mathbf{X}; \tilde{\mathbf{X}})) = \mathbf{U}^\intercal \mathbf{V}$. This allows us to write (3.2) in the explicit form:

$$\underset{\tilde{\mathbf{X}} \in \mathbb{M}}{\operatorname{argmin}} \quad R(\hat{\mathbf{X}}(\mathbf{\Theta}(\mathbf{X}; \tilde{\mathbf{X}}))) \tag{3.3}$$

In other words, we are looking for antidote data $\tilde{\mathbf{X}}$ that modifies the outputs of $\mathbf{\Theta}$ such that $\hat{\mathbf{X}}$ is modified to optimize $R$. Figure 3·1 shows a schematic representation of the antidote data effect on matrix factorization models. In the next section, we introduce an iterative method to solve (3.3).

## 3.4 Computing Antidote Data

In this section we introduce the framework for generating antidote data. We apply a projected gradient descent/ascent algorithm (`GD`/`GA`) to optimize the antidote data with respect to a socially relevant objective function. In section 3.4.1 we review a gradient descent method, introduced by Li et al. (2016), for optimizing data poisoning attacks on matrix factorization models, and which we adapt to optimize antidote data.

[1] Note that here $\mathbf{U}$ and $\mathbf{V}$ are the users and items latent vectors after adding the antidote data to the system, which can be different from initial $\mathbf{U}$ and $\mathbf{V}$.

Then, in section 3.4.2 we show how the characteristics of the antidote problem can be exploited for significant improvements in algorithmic efficiency.

### 3.4.1 A Projected Gradient Descent Approach

In this section we describe a projected gradient descent algorithm to solve the constrained optimization problem (3.2). A parallel approach is taken in (Li et al., 2016) for optimizing data poisoning attacks, which is itself an instance of the more general machine teaching problem introduced by Mei and Zhu (2015). We note that the framework introduced by Mei and Zhu (2015) can be used to extend the applicability of antidote data approach beyond matrix factorization models.

The algorithm starts from an initial antidote data with size of a given budget. At each iteration, the factorization algorithm is applied jointly on the original data and the current antidote data to find updated factors $\mathbf{U}$,$\tilde{\mathbf{U}}$,$\mathbf{V}$, and estimated ratings $\hat{\mathbf{X}}$. Then the gradient of the antidote utility with respect to antidote data at the current point is computed and the algorithm chooses a step size and updates the antidote data. After each update, a projection function is applied to get a feasible solution. In this work we only consider range constraints on the ratings, i.e., for each rating $\tilde{x}_{ij}$ we assume $\mathbb{M}_{min} < \tilde{x}_{ij} < \mathbb{M}_{max}$ where $\mathbb{M}_{min}$ and $\mathbb{M}_{max}$ indicate the minimum and maximum feasible rating in the system. Therefore the projection function simply truncates all the ratings in $\tilde{\mathbf{X}}$ at $\mathbb{M}_{min}$ and $\mathbb{M}_{max}$.

Algorithm 1 presents the details of our antidote data optimization method. If the goal is to maximize $R$, we can apply a gradient ascent algorithm by simply changing the sign of the gradient step in line 6. The learning algorithm $\mathbf{\Theta}_{\ell,\lambda}$ is an input to Algorithm 1. This is a realistic assumption in a white-box scenario, i.e., a party with the full knowledge of the recommender system seeks to generate antidote data, which is an important case. However, we emphasize that there are settings in which other parties with only partial knowledge of the system can successfully adopt the antidote data approach as well. First of all, recent work (Wang and Gong, 2018) introduces a method for estimating the hyper-parameters of a learning algorithm. Using that method we need not input $\mathbf{\Theta}_{\ell,\lambda}$ to Algorithm 1, instead only providing the original factors $(\mathbf{U}, \mathbf{V})$. Moreover, in Section 3.6 we introduce heuristic algorithms that require less information about the recommender system than does Algorithm 1.

**Algorithm 1:** Optimizing antidote data via projected gradient descent

```
Input: Observed ratings $\mathbf{X} \in \mathbb{R}^{n \times d}$, budget $n'$, factorization algorithm $\mathbf{\Theta}_{\ell,\lambda}$,
       utility $R$, feasible set $\mathbb{M}$
Output: Antidote data $\tilde{\mathbf{X}}$
Initialization initialize $\tilde{\mathbf{X}}^{(0)} \in \mathbb{R}^{n' \times d}$, $t = 0$
:
1 while convergence do
2     $\mathbf{U}, \tilde{\mathbf{U}}, \mathbf{V} = \mathbf{\Theta}(\mathbf{X}; \tilde{\mathbf{X}}^{(t)})$
3     $\hat{\mathbf{X}} = \mathbf{U}^{\intercal}\mathbf{V}$
4     Compute $\nabla_{\tilde{\mathbf{X}}} R(\hat{\mathbf{X}})$
5     Find step size $\alpha$
6     $\tilde{\mathbf{X}}^{(t+1)} = \tilde{\mathbf{X}}^{(t)} - \alpha \nabla_{\tilde{\mathbf{X}}} R(\hat{\mathbf{X}})$
7     $\tilde{\mathbf{X}}^{(t+1)} = P_{\mathbb{M}}(\tilde{\mathbf{X}}^{(t+1)})$
8     $t \leftarrow t + 1$
9 return $\tilde{\mathbf{X}}^{(t)}$
```

In order to compute $\nabla_{\tilde{\mathbf{X}}} R(\hat{\mathbf{X}})$ in line 4 of algorithm 1, we consider the explicit form of the objective function given in (3.3). Applying the chain rule we get:

$$\nabla_{\tilde{\mathbf{X}}} R(\hat{\mathbf{X}}) = \nabla_{\mathbf{\Theta}} R(\hat{\mathbf{X}}) \; \nabla_{\tilde{\mathbf{X}}}(\mathbf{\Theta}(\mathbf{X}; \tilde{\mathbf{X}})) \tag{3.4}$$

$\nabla_{\tilde{\mathbf{X}}}(\mathbf{\Theta}(\mathbf{X}; \tilde{\mathbf{X}}))$ is the Jacobian matrix that contains partial derivatives of factors $(\mathbf{U}, \tilde{\mathbf{U}}, \mathbf{V})$ with respect to each element in $\tilde{\mathbf{X}}$. These partial derivatives can be approximately computed by exploiting the KKT conditions of the factorization problem as explained in (Li et al., 2016; Mei and Zhu, 2015). However, in section 3.4.2 we show cases where the full computation of such partial derivatives is not required and we explain how to derive the necessary elements.

By applying the chain rule one more time on $\nabla_{\mathbf{\Theta}} R(\hat{\mathbf{X}})$ we get:

$$\nabla_{\tilde{\mathbf{X}}} R(\hat{\mathbf{X}}) = \nabla_{\hat{\mathbf{X}}} R(\hat{\mathbf{X}})\ \nabla_{\mathbf{\Theta}} \hat{\mathbf{X}}(\mathbf{\Theta})\ \nabla_{\tilde{\mathbf{X}}}(\mathbf{\Theta}(\mathbf{X}; \tilde{\mathbf{X}})) \tag{3.5}$$

The first term in (3.5) is the gradient of the antidote utility with respect to the estimated ratings. In this work we only consider differentiable utilities, as described in more detail in section 3.5.

The second term in (3.5) is the gradient of the estimated ratings with respect to factors $(\mathbf{U}, \tilde{\mathbf{U}}, \mathbf{V})$. This term is straighforward to compute since the ratings are linear in each factor, i.e., $\hat{\mathbf{X}} = \mathbf{U}^{\intercal}\mathbf{V}$.

In this work we do not make assumptions (e.g. convexity) about the antidote utility other than being differentiable; the framework is a general method to improve a socially relevant metric rather than one that seeks the global optimum of function $R$. However, we note that introducing antidote objectives with certain provable properties, which can provide convergence guarantees or more efficient ways to find the step size in Algorithm 1, is a potential direction for future research.

### 3.4.2 Efficient Computation of the Gradient Step

In this section we show how to further simplify (3.5) to make the update step of Algorithm 1 more efficient.

First, we write $\nabla_{\mathbf{\Theta}} \hat{\mathbf{X}}(\mathbf{\Theta})$ in terms of the block matrices that contain the partial derivatives of the estimated ratings in $\hat{\mathbf{X}}$ with respect to each factor $\mathbf{U}, \tilde{\mathbf{U}}, \mathbf{V}$, i.e.[2], $\left[\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{U}}, \frac{\partial \hat{\mathbf{X}}}{\partial \tilde{\mathbf{U}}}, \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{V}}\right]$. Notice that $\hat{\mathbf{X}} = \mathbf{U}^{\intercal}\mathbf{V}$ does not depend on $\tilde{\mathbf{U}}$ and therefore $\frac{\partial \hat{\mathbf{X}}}{\partial \tilde{\mathbf{U}}} = \mathbf{0}$.

Furthermore, we write $\nabla_{\tilde{\mathbf{X}}}(\mathbf{\Theta}(\mathbf{X}; \tilde{\mathbf{X}}))$ in terms of the block matrices that contain the partial derivatives of each factor $\mathbf{U}, \tilde{\mathbf{U}}, \mathbf{V}$ with respect to each element in $\tilde{\mathbf{X}}$, i.e., $\left[(\frac{\partial \mathbf{U}}{\partial \tilde{\mathbf{X}}})^{\intercal}, (\frac{\partial \tilde{\mathbf{U}}}{\partial \tilde{\mathbf{X}}})^{\intercal}, (\frac{\partial \mathbf{V}}{\partial \tilde{\mathbf{X}}})^{\intercal}\right]^{\intercal}$. Assuming that an infinitesimal change in $\tilde{x}_{ij}$ only results in

[2] For matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{r \times s}$, we use $\frac{\partial \mathbf{A}}{\partial \mathbf{B}}$ to denote an $mn \times rs$ matrix that contains the partial derivatives $\frac{\partial a_{ij}}{\partial b_{k\ell}}$ for each $a_{ij}$ and $b_{k\ell}$.

first order updates in vectors $\tilde{\mathbf{u}}_i$ and $\mathbf{v}_j$, we get $\frac{\partial \mathbf{U}}{\partial \tilde{\mathbf{X}}} = \mathbf{0}$.

Exploiting the fact that $\frac{\partial \mathbf{U}}{\partial \tilde{\mathbf{X}}} = \frac{\partial \hat{\mathbf{X}}}{\partial \tilde{\mathbf{U}}} = \mathbf{0}$, we can simplify (3.5) to:

$$\nabla_{\tilde{\mathbf{X}}} R(\hat{\mathbf{X}}) = \nabla_{\hat{\mathbf{X}}} R(\hat{\mathbf{X}}) \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{V}} \frac{\partial \mathbf{V}}{\partial \tilde{\mathbf{X}}} \tag{3.6}$$

Now we derive $\frac{\partial R(\hat{\mathbf{X}})}{\partial \tilde{x}_{ij}}$ for each element of the antidote data $\tilde{x}_{ij}$. Let $\mathbf{v}_1, \ldots, \mathbf{v}_d$ be the item vectors forming the columns of $V$. Then starting from the last term in (3.6) and assuming first order updates, we know that $\frac{\partial \mathbf{v}_k}{\partial \tilde{x}_{ij}}$ is non-zero only if $k = j$ and can be approximately computed as[3]:

$$\frac{\partial \mathbf{v}_j}{\partial \tilde{x}_{ij}} = (\sum_{i \in \Omega_j} \mathbf{u}_i \mathbf{u}_i^\intercal + \tilde{\mathbf{U}} \tilde{\mathbf{U}}^\intercal + \lambda \mathbf{I}_\ell)^{-1} \tilde{\mathbf{u}}_i \tag{3.7}$$

On the other hand, $\frac{\partial \hat{x}_{lk}}{\partial \mathbf{v}_j} = \mathbf{u}_l^\intercal$ if $k = j$ and an $\ell$-dimensional zero vector otherwise. Therefore, we need to compute $\frac{\partial R(\hat{\mathbf{X}})}{\partial \hat{x}_{lk}}$ only for $k = j$ and we have:

$$\frac{\partial R(\hat{\mathbf{X}})}{\partial \tilde{x}_{ij}} = \left( \sum_{l=1}^{n} \frac{\partial R(\hat{\mathbf{X}})}{\partial \hat{x}_{lj}} \frac{\partial \hat{x}_{lj}}{\partial \mathbf{v}_j} \right) \frac{\partial \mathbf{v}_j}{\partial \tilde{x}_{ij}} \tag{3.8}$$

Let $\mathbf{G}$ be a matrix formed by reshaping $\nabla_{\hat{\mathbf{X}}} R(\hat{\mathbf{X}})$ into an $n \times d$ matrix such that $g_{ij} = \frac{\partial R(\hat{\mathbf{X}})}{\partial \hat{x}_{ij}}$. Then we can write (3.8) as:

$$\frac{\partial R(\hat{\mathbf{X}})}{\partial \tilde{x}_{ij}} = \mathbf{g}_j{}^\intercal \mathbf{U}^\intercal \mathbf{S}_j^{-1} \tilde{\mathbf{u}}_i \tag{3.9}$$

where $\mathbf{S}_j = \sum_{i \in \Omega_j} \mathbf{u}_i \mathbf{u}_i^\intercal + \tilde{\mathbf{U}} \tilde{\mathbf{U}}^\intercal + \lambda \mathbf{I}_\ell$.

By using (3.9) instead of the general formula in (3.5) we can significantly reduce the number of computations required for finding the gradient of the utility function with respect to the antidote data. Furthermore, the term $\mathbf{g}_j{}^\intercal \mathbf{U}^\intercal \mathbf{S}_j^{-1}$ appears in all the partial derivatives that correspond to elements in column $j$ of $\tilde{\mathbf{X}}$ and can be precomputed in each iteration of the algorithm and reused for computing partial

[3] Details are provided in appendix A.1.

derivatives with respect to different antidote users.

## 3.5 Social Objective Functions

The previous section developed a general framework for improving various properties of recommender systems; in this section we show how to apply that framework specifically to issues of polarization and fairness.

As described in Section 3.2, polarization is the degree to which opinions, views, and sentiments diverge within a population. Recommender systems can capture this effect through the ratings that they present for items. To formalize this notion, we define polarization in terms of the variability of predicted ratings when compared across users. In fact, we note that both very high variability, and very low variability of ratings may be undesirable. In the case of high variability, users have strongly divergent opinions, leading to conflict. Recent analyses of the YouTube recommendation system have suggested that it can enhance this effect (Nicas, 2018; O'Callaghan et al., 2015). On the other hand, the convergence of user preferences, i.e., very low variability of ratings given to each item across users, corresponds to increased homogeneity, an undesirable phenomenon that may occur as users interact with a recommender system (Chaney et al., 2017). As a result, in what follows we consider using antidote data in both ways: to either increase or decrease polarization.

As also described in Section 3.2, unfairness is a topic of growing interest in machine learning. Following the discussion in that section, we consider a recommender system fair if it provides equal quality of service (i.e., prediction accuracy) to all users or all groups of users (Zafar et al., 2017b).

Next we formally define the metrics that specify the objective functions associated with each of the above objectives. Since the gradient of each objective function is used in the optimization algorithm, for reproducibility we provide the details about derivation of the gradients in appendix A.2.

### 3.5.1 Polarization

To capture polarization, we seek to measure the extent to which the user ratings *disagree*. Thus, to measure user polarization we consider the estimated ratings $\hat{\mathbf{X}}$, and we define the polarization metric as the normalized sum of pairwise euclidean distances between estimated user ratings, i.e., between rows of $\hat{\mathbf{X}}$. In particular:

$$R_{pol}(\hat{\mathbf{X}}) = \frac{1}{n^2 d} \sum_{k=1}^{n} \sum_{l>k} ||\hat{\mathbf{x}}^k - \hat{\mathbf{x}}^l||^2 \tag{3.10}$$

The normalization term $\frac{1}{n^2 d}$ in (3.10) makes the polarization metric identical to the following definition: [4]

$$R_{pol}(\hat{\mathbf{X}}) = \frac{1}{d} \sum_{j=1}^{d} \sigma_j^2 \tag{3.11}$$

where $\sigma_j^2$ is the variance of estimated user ratings for item $j$. Thus this polarization metric can be interpreted either as the average of the variances of estimated ratings in each item, or equivalently as the average user disagreement over all items.

### 3.5.2 Fairness

**Individual fairness.** For each user $i$, we define $\ell_i$, the loss of user $i$, as the mean squared estimation error over known ratings of user $i$:

$$\ell_i = \frac{||P_{\Omega^i}(\hat{\mathbf{x}}^i - \mathbf{x}^i)||_2^2}{|\Omega^i|} \tag{3.12}$$

[4] We can derive it by rewriting (3.10) as $R_{pol}(\hat{\mathbf{X}}) = \frac{1}{d} \sum_{j=1}^{d} \frac{1}{n^2} \sum_{k=1}^{n} \sum_{l>k} (\hat{x}_{kj} - \hat{x}_{lj})^2$.

Then we define the individual unfairness as the variance of the user losses:[5]

$$R_{indv}(\mathbf{X}, \hat{\mathbf{X}}) = \frac{1}{n^2} \sum_{k=1}^{n} \sum_{l>k} (\ell_k - \ell_l)^2 \tag{3.13}$$

To improve individual fairness, we seek to minimize $R_{indv}$.

**Group fairness.** Let $I$ be the set of all users/items and $G = \{G_1 \ldots, G_g\}$ be a partition of users/items into $g$ groups, i.e., $I = \bigcup_{i \in \{1,\ldots,g\}} G_i$. We define the loss of group $i$ as the mean squared estimation error over all known ratings in group $i$:

$$L_i = \frac{||P_{\Omega_{G_i}}(\hat{\mathbf{X}} - \mathbf{X})||_2^2}{|\Omega_{G_i}|} \tag{3.14}$$

For a given partition $G$, we define the group unfairness as the variance of all group losses:

$$R_{grp}(\mathbf{X}, \hat{\mathbf{X}}, G) = \frac{1}{g^2} \sum_{k=1}^{g} \sum_{l>k} (L_k - L_l)^2 \tag{3.15}$$

Again, to improve group fairness, we seek to minimize $R_{grp}$.

### 3.5.3 Accuracy vs. Social Welfare

Adding antidote data to the system to improve a social utility will also have an effect on the overall prediction accuracy. Previous works have considered social objectives as regularizers or constraints added to the recommender model (eg, (Burke et al., 2018; Zafar et al., 2017d; Kamishima et al., 2011)), implying a trade-off between the prediction accuracy and a social objective.

However, in the case of the metrics we define here, the relationship is not as simple. Considering polarization, we find that in general, increasing or decreasing polarization will tend to decrease system accuracy. In either case we find that system accuracy only declines slightly in our experiments; we report on the specific values

[5] Note that for a set of equally likely values $x_1, \ldots, x_n$ the variance can be expressed without referring to the mean as: $\frac{1}{n^2} \sum_{i} \sum_{j>i} (x_i - x_j)^2$.

in Section 3.6. Considering either individual or group unfairness, the situation is more subtle. Note that our unfairness metrics will be exactly zero for a system with zero error (perfect accuracy). As a result, it is possible that as the system decreases unfairness, overall accuracy may either increase or decrease. We illustrate these effects in our experiments in Section 3.6.

## 3.6 Effectiveness

In this section we use the tools developed in previous sections to study the effectiveness of antidote data in varying the polarization and reducing the unfairness of a matrix-factorization based recommender system.

We consider a recommender system that estimates unknown ratings by solving the regularized matrix factorization problem as defined by (3.1). We implemented an alternating least squares algorithm (Hastie et al., 2015; Hardt, 2014) to find the factors. We use the MovieLens 1M dataset which contains around 1 million ratings of ~4000 movies made by ~6000 users, with ratings on a 5-point scale (Harper and Konstan, 2016). We choose the 1000 most frequently rated movies, and use different subsets of users in different experiments as described below.

For each dataset we perform a validation process to choose the hyper-parameters $(\ell, \lambda)$ so as to obtain realistic settings. The hyper-parameters are selected based on the average root-mean-square error (RMSE) of the factorization in multiple random splits of observed ratings into training and validation sets. We assume that the hyper-parameters are fixed during the antidote data generation process since the antidote data is generated for a fixed recommender system.

First we show the effectiveness of antidote data in modifying the user polarization as defined in section 3.5.1. In section 3.6.2 we describe different heuristics that can significantly speed up the construction of antidote data. Finally, section 3.6.3

demonstrates the effectiveness of applying antidote data for improving fairness.

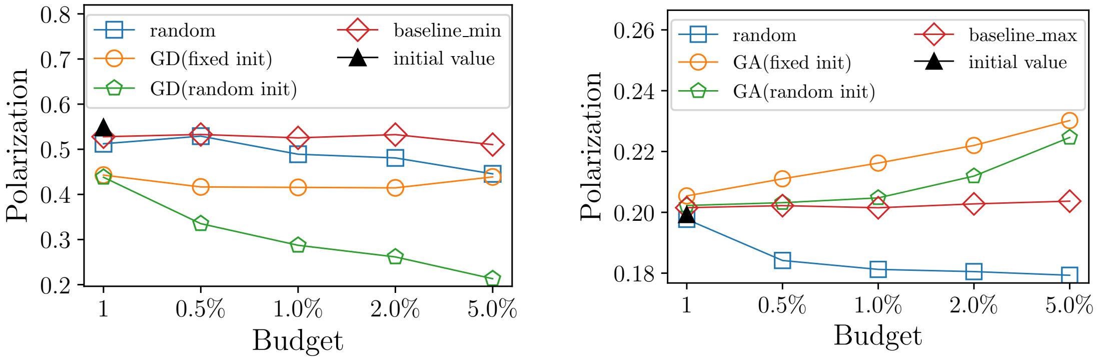
### 3.6.1 Polarization

To explore modifying user polarization, we choose a random subset of 1000 users yielding a matrix in which 11% of the elements are known. As previously mentioned, it may be of interest to either increase or decrease the polarization metric in different scenarios. We present an example for each case. We do so by taking advantage of the fact that different hyperparameter combinations can yield models that are very close in overall accuracy but that differ significantly with respect to initial user polarization in the system.

In particular, we observe that the average validation RMSE over ten random splits of observed ratings into training and validation sets for $(rank = 8, \lambda = 10)$ is **0.87** and for $(rank = 4, \lambda = 0.1)$ is **0.90**. However, the polarization $(R_{pol})$ of the estimated rating matrix for $(rank = 8, \lambda = 10)$ is 0.2 whereas for $(rank = 4, \lambda = 0.1)$ the polarization goes up to 0.55. We use the former setting as an example where the goal is to increase the polarization metric (to avoid homogeneity), and the latter setting as an example of a polarized system where the goal is to reduce polarization.

For each of the maximization and minimization objectives, we compare the performance of the antidote data generation framework with a baseline algorithm. When seeking to minimize polarization, we use `baseline_min`. This algorithm tries to reduce the variance of estimated ratings in each item by setting the ratings given to the corresponding item in the antidote data to the average of known ratings for that item in the original data. When seeking to maximize polization, we use `baseline_max.` This algorithm generates antidote data by setting half of the user ratings in each item to the maximum feasible rating value and the other half of the ratings to the minimum feasible rating value.

Furthermore, we consider two different initializations for the optimization process:



**(a)** Minimizing polarization

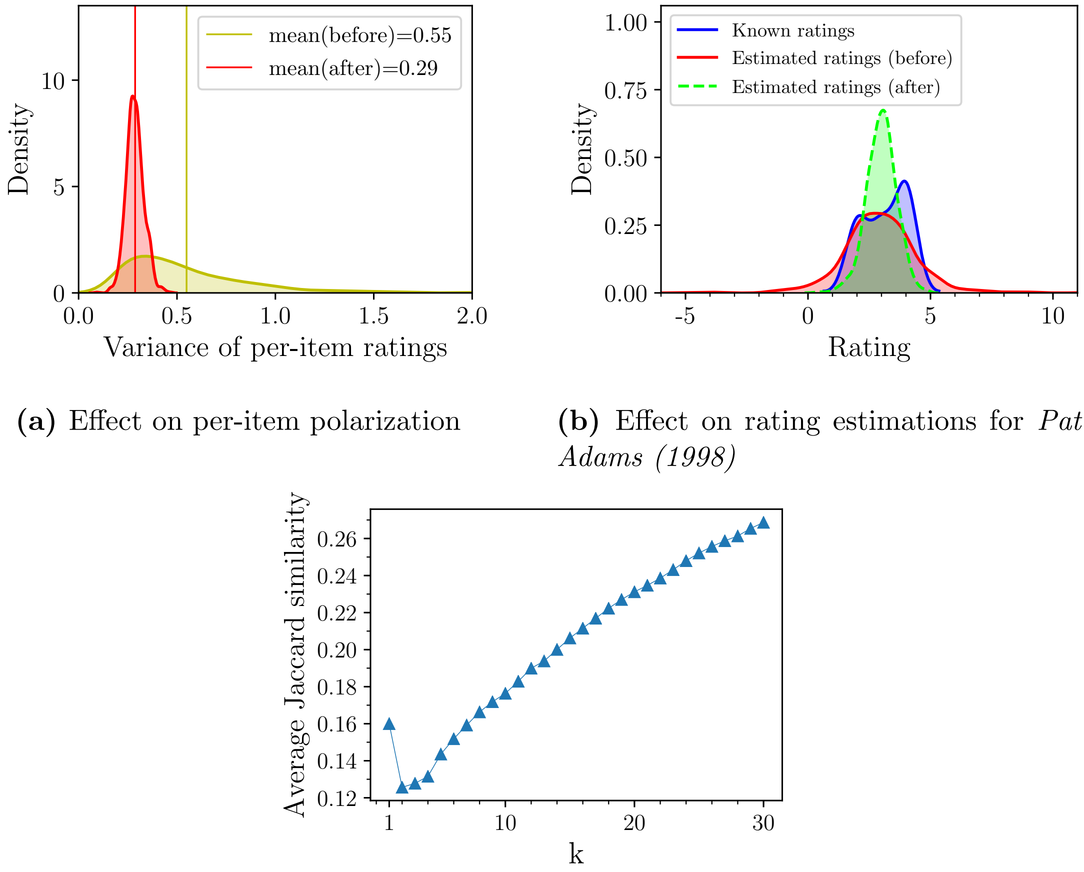
**(b)** Maximizing polarization

**Figure 3·2:** Modifying user polarization.

in the case of `GD(fixed init)`, all the ratings in the initial antidote data are set to the same value. In the case of `GD(random init)`, we run the optimization multiple times starting from random initializations and return the best solution.

Figure 3·2 compares the effects of adding antidote data constructed by different methods on polarization. After each injection of the antidote data, the new polarization is computed using the original data only, i.e., we ignore the injected data in evaluating polarization. We present our results for different budgets varying from a single antidote user to 5% of the number of original users. We also show the effect of ratings that are randomly generated over the feasible range, when used as antidote data.

Our results show that the antidote data generation framework can successfully either minimize or maximize polarization. Antidote data generated by our method are considerably more effective than the baseline algorithms as well as random data. We observe that a 2% budget is enough to reduce the initial polarization in a polarized setting by 50% and increase the polarization in a less polarized setting by 10%. Furthermore, we observe that random initialization is more effective for minimizing polarization whereas initializing all the antidote ratings from the same value is more effective for maximizing polarization.



**(a)** Effect on per-item polarization

**(b)** Effect on rating estimations for *Patch Adams (1998)*

**(c)** Effect on top-k recommended items

**Figure 3·3:** Minimizing polarization with 1% budget.

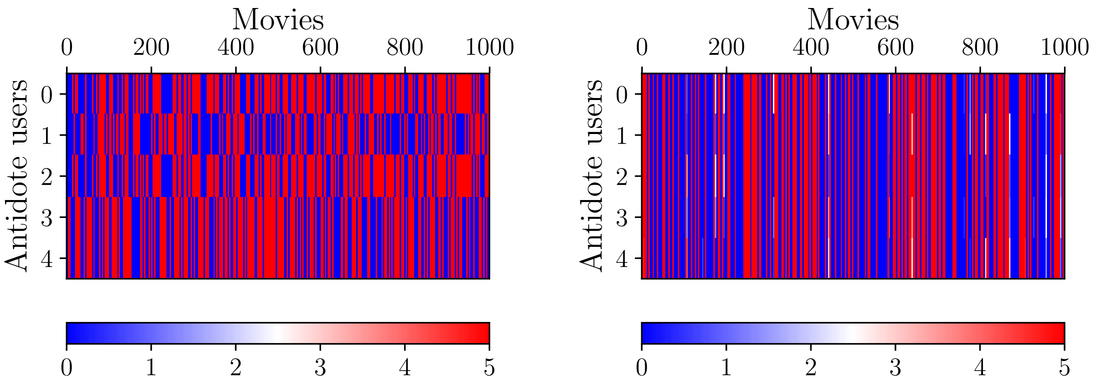
To better understand the effect of antidote data on user polarization, in Figure 3·3 we demonstrate the effect of antidote data with a 1% budget for the minimization case. Note that the effect on estimation error of adding antidote data is negligible: RMSE of rating estimations for known elements changes from **0.80** to **0.83**. In other words, antidote data modifies the prediction model such that its predictions still approximately agree with the known ratings but the polarization of the new estimated rating matrix is significantly different.

Figure 3·3a shows the distributions of per-item polarization ($\sigma_j^2$ in (3.11)) along with $R_{pol}$ (the distributional mean) before and after antidote data injection. The figure shows that without antidote data, a small set of items make large contributions to overall polarization – they have quite high variance in ratings, shown by the long distributional tail. The addition of antidote data dramatically reduces this effect, and also significantly lowers $R_{pol}$ from 0.55 to 0.29.

Figure 3·3b shows the effect of adding antidote data on the estimated ratings of *Patch Adams (1998)*, one of the movies for which the variance of estimated ratings is large before adding antidote data. We observe that the distribution of known ratings for this movie indicates a polarized case with two peaks at 2 and 4. The initial rating estimations in this case lie in an interval that is much larger than the range of observed ratings. Adding antidote data modifies the extreme rating estimations; resulting in a unimodal distribution over the range of original ratings.

While the goal of adding antidote data is to modify the system's predicted ratings, an important use case for such a system is to output the top rated items as the system's recomendations. Hence, it is important to ask how modifying predicted ratings will change the ranking of unrated items, i.e., the output of a top-$k$ recommender system. Therefore, we consider the top-$k$ recommended items on a per-user basis and measure the degree of change in the recommendations before and after adding antidote data. We use the Jaccard similarity of the sets of recommended items to measure this change.

Figure 3·3c shows the average of Jaccard similarities across all users. Our results show that the antidote data significantly changes the output of a top-$k$ recommender system. For example, adding 1% additional antidote data changes the top-recommended item for 84% of all users. We observe that, in general, as the number of considered top items grows, the effect lessens (Jaccard similarity grows). However, the changes in the set of recommended items are still significant up to $k = 30$.



**(a)** Minimizing polarization

**(b)** Individual fairness

**Figure 3·4:** Optimal antidote data with 0.5% budget.

### 3.6.2 Heuristic Algorithms

In this section we introduce heuristic algorithms that dramatically reduce the computational cost of antidote data generation. Notice that the computational cost of Algorithm 1 is dominated by performing the matrix factorization algorithm (evaluating $\mathbf{\Theta}$) in each pass through the gradient descent loop. The heuristics are designed based on various approximations that can be made in different steps of Algorithm 1 to minimize the number of times $\mathbf{\Theta}$ is evaluated. The approximations are motivated by certain patterns observed in the antidote data generated by Algorithm 1.

Figure 3·4 shows the antidote data generated by `GD(random init)` for minimizing polarization (Fig. 3·4a) and minimizing individual unfairness (Fig. 3·4b). We observe that: (i) most of the ratings in the resulting antidote data are equal to one of the boundary values in the feasible set, $\mathbb{M}_{min}$ or $\mathbb{M}_{max}$ (0 or 5 in our experiments), and (ii) in the fairness case, most of the users (rows) in the antidote data converge to a nearly-identical pattern of ratings over items, even if they are initialized with different random values.

Based on the above observations, in our evaluations we consider two heuristics for generating antidote data for fairness. The first (`heuristic1`) offers considerable

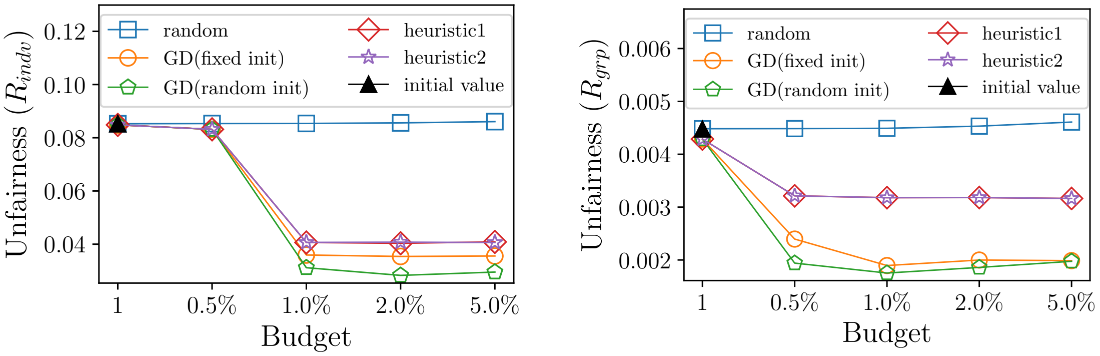
**Table 3.1:** Effect of antidote data on individual unfairness $R_{indv}$ in the held-out ratings. $R_{indv}$ before antidote is 0.1087.

| | Budget | | | | |
|---|---|---|---|---|---|
| Algorithm | 1 | 0.5% | 1% | 2% | 5% |
| GD(random init) | 0.1086 | 0.1084 | 0.1054 | 0.1157 | 0.1086 |
| GD(fixed init) | 0.1086 | **0.1083** | 0.0929 | 0.0985 | 0.0968 |
| heuristic1 | 0.1086 | 0.1084 | **0.0816** | **0.0800** | 0.0830 |
| heuristic2 | 0.1086 | 0.1084 | 0.0817 | 0.0818 | **0.0811** |

computational savings, and the second (`heuristic2`) offers even more savings, while additionally removing the need for access to the factorization algorithm or its hyper-parameters ($\ell$,$\lambda$)[6].

`heuristic1` reduces the number evaluations of $\mathbf{\Theta}$ to a single call by combining observations (i) and (ii). It works by considering the addition of only a single row of antidote data, and computes gradients for that row. Rather than performing gradient descent over a series of small steps, it then simply sets each value in the antidote data row to either $\mathbb{M}_{min}$ or $\mathbb{M}_{max}$ depending on the sign of the gradient. It then replicates the resulting row as many times as dictated by the antidote data budget.

In the case of `heuristic2`, in addition to using the above observations, we approximate the direction of the gradient without the need to perform matrix factorization, given access to factors $\mathbf{U}$ and $\mathbf{V}$. In this case, access to the factorization algorithm or its hyper-parameters is not required. Notice that (3.9) can be rewritten as $\frac{\partial R(\hat{\mathbf{X}})}{\partial \tilde{x}_{1j}} = \mathbf{a}_j^\intercal \mathbf{b}_j$ where $\mathbf{a}_j = \left[\mathbf{g}_j^\intercal \mathbf{U}^\intercal\right]^\intercal$ and $\mathbf{b}_j = \mathbf{S}_j^{-1}\tilde{\mathbf{u}}_1$. For sufficient level of regularization $\lambda$, we can approximate $\mathbf{a}_j^\intercal \mathbf{b}_j \approx c\mathbf{a}_j^\intercal \mathbf{1}_\ell$ where $c$ is a constant and $\mathbf{1}_\ell$ is an $\ell$-dimensional vector of 1's. This leads to a modification of `heuristic1` in which all the values in column $j$ of the antidote data are set to $\mathbb{M}_{min}$ or $\mathbb{M}_{max}$ depending on the sign of $\mathbf{g}_j^\intercal \mathbf{U}^\intercal \mathbf{1}_\ell$.



**(a)** Individual fairness

**(b)** Group fairness

**Figure 3·5:** Improving fairness.

### 3.6.3 Fairness

In this section we show how antidote data as generated by our various algorithms improves fairness. We again use the MovieLens dataset; to study group fairness, we group movies by genre as specified in the dataset. In contrast to the case for polarization, the fairness objective is a function of both the known and predicted user ratings. Hence we choose the 1000 most active users and the 1000 most frequently rated movies. This gives us a rating matrix in which ∼36% of the elements are known. For this dataset we run the matrix factorization algorithm with hyper-parameters $(rank = 8, \lambda = 1)$.

To verify that adding antidote data improves the fairness of unseen ratings, we hold out 20% of the known ratings per user as a test set. We use the remaining data (training set) to generate antidote data; we then measure the effectiveness of the resulting antidote data in both training and test sets.

We start by assessing the effect of antidote data on fairness in the training data. We show the impact of antidote data on individual unfairness ($R_{indv}$) in Figure 3·5a and on group unfairness ($R_{grp}$) in Figure 3·5b. The figures compare the effect of

[6] Pseudocodes are provided in appendix B.

**Table 3.2:** Effect of antidote data on group unfairness $R_{grp}$ in the held-out ratings. $R_{grp}$ before antidote is 0.0088.

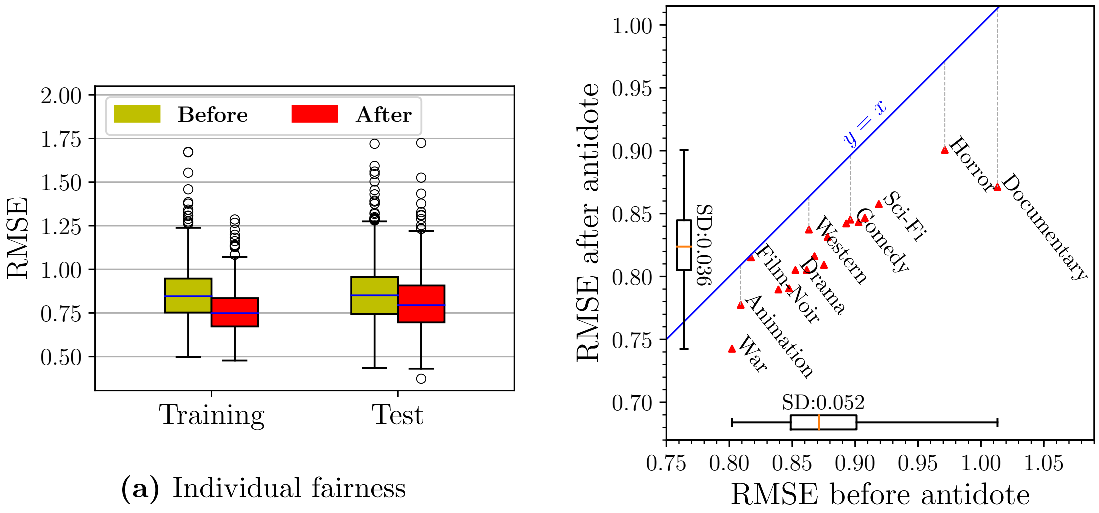
| | Budget | | | | |
|---|---|---|---|---|---|
| Algorithm | 1 | 0.5% | 1% | 2% | 5% |
| GD(random init) | 0.0087 | **0.0035** | 0.0041 | 0.0042 | 0.0045 |
| GD(fixed init) | 0.0087 | 0.0042 | **0.0040** | **0.0040** | **0.0044** |
| heuristic1 | 0.0087 | 0.0055 | 0.0056 | 0.0057 | 0.0058 |
| heuristic2 | 0.0087 | 0.0055 | 0.0056 | 0.0057 | 0.0058 |

antidote data as generated by four different algorithms: Algorithm 1 with two different initializations as described in Section 3.6.1, and the two heuristic algorithms introduced in Section 3.6.2.

The results show that all algorithms improve fairness considerably. In fact most of the benefits of antidote data can be obtained by only adding 1% additional users in the individual fairness case and 0.5% additional users in the group fairness case. The figures also show that the much simpler heuristics, in which all rows of the antidote data are identical, are effective: for individual fairness, they provide almost all the benefits of Algorithm 1 while for group fairness they provide around half of the benefits of Algorithm 1.

Tables 3.1 and 3.2 show the resulting values of the individual and group unfairness metrics in the *test set* after antidote data addition for different budgets and different algorithms. We observe that the antidote data generated to reduce unfairness in the training data is also effective for reducing unfairness on the the held-out test data. The optimal value (minimum unfairness) in each table is highlighted. We observe that even in the test set, a 2% budget using `heuristic1` can reduce individual unfairness by over 25% (from 0.1087 to 0.0800), and group unfairness can be lowered by more than 50% (from 0.0088 to 0.0035) using `GD(random init)` and a 0.5% budget.

Figure 3·6 provides more insight into how adding antidote data reduces individual and group unfairness. In each case, we consider the setting that reaches the minimum unfairness in the test set as presented in tables 3.1 and 3.2.



**(a)** Individual fairness

**(b)** Group fairness

**Figure 3·6:** Antidote data effect on fairness.

Figure 3·6a shows the effect of optimal antidote data on per-user RMSEs. The figure demonstrates a number of points. First, adding antidote data results in a model with less variation in per-user RMSE of rating estimations in both training and test sets. Second, a noticeable way in which adding antidote data improves fairness is by reducing the magnitude of the outliers that drive unfairness in both training and testing. Finally, the figure shows that in this example adding antidote data actually improves overall accuracy of the model predictions.

Figure 3·6b shows the effect of optimal antidote data on per-group RMSE in the test set. For each group (genre) of movies, the corresponding point shows the group's RMSE before and after adding antidote data. Additionally, the boxplots on each axis illustrate the distribution of RMSE values across groups before and after adding antidote data.

First, we observe that all points are below the line $y = x$, i.e., adding antidote data improves the prediction accuracy of all genres and thus the overall accuracy of the model. Moreover, the boxplots show that improvements in rating estimations are so that the cross-group variability in RMSE is decreased to reach a fairer situation. Finally, we see that outliers particularly benefit from addition of antidote data; this can be seen as larger RMSE improvements in genres that initially had larger RMSE. In particular, *Documentary* and *Horror* have the largest prediction errors before adding antidote data, and their RMSEs are the most improved (furthest below $y = x$) after adding antidote data.

## 3.7 Summary

In this chapter we proposed a new strategy for improving the socially relevant properties of a recommender system: adding antidote data. We have presented an algorithmic framework for this strategy and applied it to a range of socially important objectives. Using this strategy, one does not need to modify the original system input data or the system's algorithm. We show that the resulting framework can efficiently improve the polarization or fairness properties of a recommender system. We conclude that the developed framework can be a flexible and effective approach to addressing the social impacts of a recommender system.

# Chapter 4

# Fair Inputs and Fair Outputs

## 4.1 Introduction

As data-driven decision making systems are increasingly used in modern society in ways that affect individual lives, concerns have been raised about their ethical implications. In particular, recent years have witnessed a fast-growing number of studies on fairness in the decisions made by such systems, including works on developing notions to define, measures to quantify, and mechanism to ensure *fair outputs* (i.e., whether a decision system provides an equitable service to all of its users or groups of users). Despite the natural dependence of decision outcomes on data inputs, fairness concerns that incorporate the *inputs* of decision system are however less studied.

Traditionally, ethical concerns about the *inputs to* (i.e., data used by) decision systems have been the focus of "privacy" studies, while ethical concerns about the *outputs from* decision systems have been the focus of "fairness" studies. However, we observe that privacy and fairness originate from fundamentally different epistemic arguments. At a high-level, privacy concerns are rooted in a desire to protect individuals by limiting or enabling control over the information they reveal to the world. Fairness concerns, on the other hand, are rooted in a desire for equitable treatment of individuals (or groups of individuals). As such, privacy and fairness concerns can independently arise for both the inputs used and the outputs generated by decision systems.

As a motivating example, consider a decision problem where the goal is to decide whether an applicant should be offered a loan. The decision for each applicant is made based on answers that are collected to a number of demographic and financial status questions. In settings similar to this example, we recognize certain social concerns regarding the information that is gathered from each applicant. In particular, we raise two questions motivated by previous proposals and legal regulations:

First, considering each applicant individually, we ask *"what information is necessary for (i.e., what questions are relevant to) solving the decision problem at hand?"* In the loan eligibility problem for example, it seems unnecessary to ask about an applicant's height. Moreover, although it may often be necessary to ask about education level, an applicant who has an excellent credit score and a secure job may find a question about his education level to be unnecessary. Notice that as revealing each piece of information to the decision system is associated with a potential loss in privacy, this concern is related to protecting individuals' privacy.

The above consideration is reflected in the EU General Data Protection Regulation (GDPR) (Regulation, 2016) as a principle called *data minimization*, which is defined as: *"Personal data shall be adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed."*

The second ethical question arises when comparing the information used (i.e., set of questions asked) from different applicants. In particular, we ask *"how can using different pieces of information from different applicants amount to discrimination?"* For instance, a loan applicant may find it unfair that she is asked to answer a different set of questions comparing to another applicant.

In order to study these questions in a concrete setting, we consider a classifier and a set of input variables (features). We assume that the classifier is trained using all the features, and we study properties of the classifier when it is applied to a test set. Furthermore, we assume that the classifier is able to classify a given data point using any subset of the input features. In other words, for a given classification instance at the test time, the values of only a subset of input features may be revealed to the classifier, and the remaining feature values are set to *unknown*[1].

We observe that in such scenarios, one may ask *"What properties can make the set of data inputs used for each classification instance more socially desirable?"* Our first contribution is proposing two properties of classifiers regarding their inputs to address the above question; namely the *need-to-know principle* and the *fair privacy principle*. In the following we introduce each principle and their formal definitions will be presented in section 4.4.

Notice that we are not concerned with *how* a set of feature values are selected to be used for classifying each instance, but we are rather interested in checking whether an arbitrary set of features meets these properties[2].

**The need-to-know principle.** This property presents one way of formalizing *"data minimization"* (Regulation, 2016) in a classification setting. We propose a formulation that is based on classification accuracy. Intuitively, the need-to-know principle requires that the decision system use only *the minimal amount of information* that is necessary for classifying a data point with a certain accuracy. This may for example result in restricting the use of irrelevant or proxy features.

The justification for this principle is rooted in respecting the privacy rights of individuals to not divulge information about themselves that is not needed for the task at hand. Such a consideration is an important argument for emerging privacy

[1] While we do not make additional assumptions about the classification algorithm, practical examples of classification with partially known inputs are using models that can handle different sets of input features (e.g. the naive Bayes), or using some imputation procedure to estimate unknown feature values.

[2] In practice, one needs to specify how the features are selected (e.g., by using methods that are suggested in Shim et al. (2018); Maliah and Shani (2018); Trapeznikov and Saligrama (2013); Yang and Honavar (1998)). However, by studying these properties and their interaction regardless of the feature selection procedure, we show inherent trade-offs that cannot be avoided using any feature selection procedure.

regulations in different countries that require data aggregators to justify the need to collect information about individuals (Regulation, 2016; Reidenberg, 1994; of Health et al., 2003).

**The fair privacy principle.** Intuitively, the fair privacy principle requires that the decision system use the *same information (i.e., data inputs)* about all individuals when making decisions. Put differently, the fair privacy principle prohibits a decision system from using more or less or different pieces of information about different individuals. The justification for the fair privacy principle is two-fold:

First, we observe that in many scenarios it is preferable to use the same data inputs for all individuals since it equalizes the opportunity to get beneficial outcomes. In the loan eligibility problem for example, if a decision system uses different input features for two different individuals say, Alice and Bob, Alice might wonder if she might have been offered a loan had she been asked to provide the same inputs as Bob, and vice versa.

We do not expect this argument to be desirable in every situation. For example, in the case of predicting recidivism rates, it may seem reasonable to ask for more information from one individual comparing to the others in order to achieve an accurate prediction. However, in other domains such as recruiting, it is often considered best practice for all candidates to be asked the same questions, i.e., provide the same data inputs. In fact such considerations have been the main inspiration for *structured interviews.*

Second, note that one approach to achieve “equitable treatment of individuals”— as the basic idea behind fairness— is equitable protection of individuals against disclosure of their private data. Ekstrand et al. (2018) suggest that a desirable property for a privacy protection mechanism is to provide its protections equitably to all its subjects. From this perspective, In decision scenarios where individuals would prefer to not divulge their private data and there is cost to revealing such data, it is preferable that all individuals bear equal cost, and our fair privacy principle guarantees such an equatable share of privacy costs.

Our running assumption in this chapter is that the decision system (classifier) itself is a privacy adversary. This assumption is consistent with scenarios such as our loan application example. Thus we do not consider privacy notions that assume an adversary who is different from the party that collects and processes personal data (e.g., differential privacy (Dwork et al., 2014)).

**The trade-off.** Our second contribution lies in exposing the trade-offs in simultaneously achieving the proposed fairness and privacy considerations for inputs as well as previously proposed fairness considerations for outputs. Specifically, after formalizing our proposed principles of need-to-know and fair privacy, we show that in general, an *optimal* classifier cannot simultaneously satisfy both principles and achieve fairness in outputs (defined as equal prediction accuracy for all individuals). We then provide a formal specification of all datasets in which this trade-off exists, and a practically efficient algorithm to verify whether a given dataset presents the trade-off.

While each of need-to-know and fair privacy is a desirable property by itself and it is natural to seek a classifier that satisfies both, we further explain why achieving these two properties simultaneously is particularly interesting. Assume a classifier is applied to solve our loan eligibility example. One may decide to achieve fair privacy by asking all applicants to provide answers to all the input features. While this trivial approach will satisfy fair privacy, we observe that for all applicants whose prediction would not change if using a subset of feature values, the need-to-know principle is violated[3].

[3] Another solution is to use a trivial classifier that does not use any feature values from all applicant. Notice that this trivial classifier violates the adequacy requirement of input data in the "data minimization" principle.

Therefore, imposing need-to-know constraint can be seen as a way to eliminate trivial solutions for achieving fair privacy. On the other hand, using the same subset of input features for all the applicants such that need-to-know is respected will affect the prediction accuracy of those applicants for whom more data inputs are required. Our incompatibility result in fact formalizes this intuitive argument.

Finally, note that although optimal classifiers are rarely used in practice, our results pose a new challenge to the design of classifiers that aim at optimality: "how much one needs to compromise on optimality in order to simultaneously achieve fairness in the inputs and outputs of a classifier?"

## 4.2 Related Work

While some recent work has focused on both privacy and fairness considerations for outputs (Cummings et al., 2019; Feldman et al., 2015; Melis et al., 2019; Bagdasaryan et al., 2019; Jagielski et al., 2019), relatively little work (e.g., (Grgic-Hlaca et al., 2018)) has examined fairness considerations for inputs. In this Chapter we introduced new notions that simultaneously capture both privacy and fairness properties of inputs in algorithmic decision systems, and explore their interaction with fairness properties of outputs. In the following we review more related work on different societal aspects of decision-making systems including privacy and fairness. For a general overview of fairness in machine learning see Section 2.1.

**Cost-sensitive learning and privacy as cost.** Our definitions of fairness in privacy can be expressed in terms of a cost associated with each feature. This follows a line of research in machine learning that focuses on settings in which acquiring feature values is associated with some cost. The goal then is to make the best possible prediction with minimum cost users incurred at the test time. Some examples are decision trees with minimal cost (Ling et al., 2004), test-cost sensitive Naive Bayes classification (Chai et al., 2004), and using a Markov decision process to sequentially acquire feature values (Shim et al., 2018; Maliah and Shani, 2018; Trapeznikov and Saligrama, 2013).

A number of previous papers have associated privacy more explicitly with a cost (Pattuk et al., 2015; Early et al., 2016). Note however that while these works consider privacy as feature costs, the general goal is that the privacy loss of each individual is minimized; there is no consideration of fairness of privacy.

**Privacy and fairness.** Recently both privacy and fairness researchers have recognized the importance of understanding the interaction between privacy and fairness in algorithmic decision systems (Cummings et al., 2019; Jagielski et al., 2019; Bagdasaryan et al., 2019; Hajian et al., 2015; Ruggieri et al., 2014). However, as eloquently argued by Ekstrand et al. (2018), much work remains to be done in "characterizing under what circumstances and definitions privacy and fairness are simultaneously achievable?". Our results in this chapter can be seen as an effort to answer this question by specifying some of the circumstances in which the interaction between privacy and fairness can be formally studied.

Ekstrand et al. (2018) also interpret fair privacy as whether a privacy scheme protects all individuals equally, and they raise questions about the implications of this property on other fairness notions; however their discussion remains at a high level.

From a practical viewpoint, some studies have proposed techniques to improve fairness and privacy at the same time (Hajian et al., 2015, 2012; Ruggieri et al., 2014). Furthermore, Pratesi et al. (2018, 2020) provide a framework for empirical assessment of privacy risks associated with different individuals when different subsets (dataviews) of a dataset are used. This framework allows studying the trade-off between privacy risk and data utility (which in turn is linked to accuracy). However, they do not consider an explicit notion of fairness in privacy or in accuracy.

**Differential Privacy and fairness.** Another recent line of work considers a common pivacy notion, differential privacy (Dwork et al., 2014), and studies its interaction with existing fairness notions. In particular, Cummings et al. (2019) prove that differential privacy is incompatible with satisfying equal false negative rates among groups, and they provide a differentially private classification algorithm that approximately satisfies group fairness guarantees with high probability. Furthermore, it has been shown that applying differential privacy implies unequal accuracy costs over different subgrups which results in decreasing fairness (Bagdasaryan et al., 2019).

In this chapter, instead of differential privacy, we use a privacy notion that is based on the set of revealed features of users, which allows us to compare the privacy loss of different users.

**Incompatibility results.** There are incompatibility results in the area of fairness in machine learning (Chouldechova, 2017; Kleinberg et al., 2016); however, they all consider different fairness measures defined for the output of a learning system, e.g., the trade-off between calibration, equal false positive rates, and equal false negative rates. In contrast, this chapter introduces a trade-off between fairness properties related to the outputs and inputs of a classifier.

## 4.3 Formulation and Setting

We start by establishing notation and a number of definitions. We consider a set of features $F = \{f_1, \ldots, f_d\}$ in which each feature $f_i$ takes values from the domain $\mathcal{F}_i$. A dataset $\mathcal{D}$ is a set of data points (feature vectors) $\mathbf{x}_i \in \mathcal{X}$ where $\mathcal{X} = \mathcal{F}_1 \times \cdots \times \mathcal{F}_d$ together with the corresponding labels $y_i \in \mathcal{Y}$, i.e., $\mathcal{D} \subset \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$. For notational convenience, we use $\mathcal{D}_{\mathcal{X}}$ to denote the set of feature vectors in $\mathcal{D}$, i.e., $\mathcal{D}_{\mathcal{X}} = \{\mathbf{x}_i | \exists y \in \mathcal{Y} \ s.t. \ (\mathbf{x}_i, y) \in \mathcal{D}\}$. For any $S \subseteq F$ and $\mathbf{x}_i$, $\Omega_S(\mathbf{x}_i)$ denotes feature vector $\mathbf{x}_i$ in which only values of the features in $S$ are revealed.

Let $X$ be a multivariate random variable that takes on values $\mathbf{x} \in \mathcal{D}_{\mathcal{X}}$, and $Y(X)$ be a random variable that denotes the true label of $X$ in $\mathcal{D}$. If no information about $X$ is known, the probability that the label of $X$ is $c \in \mathcal{Y}$ equals to[4]

$$Pr[Y(X) = c] = \frac{|\{(\mathbf{x}, y) \in \mathcal{D} | y = c\}|}{|\mathcal{D}|}$$

This probability changes if some features values in $X$ are revealed. In particular, given that $\Omega_S(X) = \Omega_S(\mathbf{x}_i)$ we have:

$$\begin{aligned} Pr[Y(X) =& c \,|\, \Omega_S(X) = \Omega_S(\mathbf{x}_i)] = \\ & \frac{|\{(\mathbf{x}, y) \in \mathcal{D} | \Omega_S(\mathbf{x}) = \Omega_S(\mathbf{x}_i) \wedge y = c\}|}{|\{(\mathbf{x}, y) \in \mathcal{D} | \Omega_S(\mathbf{x}) = \Omega_S(\mathbf{x}_i)\}|} \end{aligned}$$

A classifier $\hat{Y}$ is a function that predicts the label of a given feature vector. We assume that $\hat{Y}$ is trained on all the features in $F$, and at the test time it is applied to data points in a dataset $\mathcal{D}$. Furthermore, We assume that $\hat{Y}$ can make a prediction using any subset of the feature values (see footnote 1). In particular, $\hat{Y}(\Omega_S(\mathbf{x}_i))$ denotes the predicted label for $\mathbf{x}_i$ by $\hat{Y}$ using feature set $S$. We do not make any assumption about $\hat{Y}$ being a deterministic or a probabilistic function.

$\hat{Y}(X)$ is a random variable that denotes the label predicted for $X$ by $\hat{Y}$; similarly, $\hat{Y}(\Omega_S(X))$ is a random variable that denotes the label predicted for $X$ by the classifier $\hat{Y}$ based on the features in $S$.

### 4.3.1 Predictive Power of a Feature Set

For a given dataset $\mathcal{D}$, we define the *predictive power* $\Phi_S(\mathbf{x}_i)$ of a feature set $S \subseteq F$ for a data point $\mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}$, as the probability of the most probable label for $X$ given

[4] In this chapter we assume a finite sample model using the given dataset. Thus our setting is an instance of transductive learning as opposed to inductive learning in which the dataset is a sample from some distribution.

that the values of the features in $S$ are revealed by $\mathbf{x}_i$, i.e., $\Omega_S(X) = \Omega_S(\mathbf{x}_i)$. In other words,

$$\Phi_S(\mathbf{x}_i) = \max_{c \in \mathcal{Y}} Pr[Y(X) = c | \Omega_S(X) = \Omega_S(\mathbf{x}_i)]$$

If $\Phi_S(\mathbf{x}_i) = 1$, we say that $\mathbf{x}_i$ is distinguishable in $\mathcal{D}$ using feature set $S$.

### 4.3.2 Optimal Classifier

We first define the accuracy of a classifier for a data point using a subset of features.

**Prediction Accuracy.** The *accuracy* of the prediction $\hat{Y}(\Omega_S(\mathbf{x}_i))$ is the probability that the label predicted for $X$ using the features in $S$ is equal to the true label of $X$, given the feature values revealed by $\Omega_S(\mathbf{x}_i)$. In other words,

$$acc(\hat{Y}(\Omega_S(\mathbf{x}_i))) = Pr[\hat{Y}(\Omega_S(X)) = Y(X) | \Omega_S(X) = \Omega_S(\mathbf{x}_i)]. \tag{4.1}$$

An optimal classifier is then defined as follows.

**Optimal Classifier.** Given a dataset $\mathcal{D}$, an optimal classifier $\hat{Y}_{opt}$ is a classifier that for all data points in $\mathcal{D}$ and using any subset of features $S \subseteq F$, has the highest prediction accuracy. In other words, $\hat{Y}_{opt}$ satisfies the following[5]

$$\forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \forall S \subseteq F, \forall \hat{Y};\ acc(\hat{Y}(\Omega_S(\mathbf{x}_i))) \leq acc(\hat{Y}_{opt}(\Omega_S(\mathbf{x}_i)))$$

The following lemma provides a convenient way for computing the accuracy of the predictions made by an optimal classifier. In particular, it states that for any data point in a given dataset, the accuracy of an optimal classifier using a set of features can be computed by finding the predictive power of that feature set for the corresponding data point. We later use this result to measure the performance of an optimal classifier by studying the characteristics of the dataset to which the classifier is applied.

[5] This is an extension of the *Bayes optimal classifier* to the settings where any subset of features can be used to make a prediction.

**Lemma 1.** *A classifier is optimal for a given a dataset $\mathcal{D}$, if and only if for any $\Omega_S(\mathbf{x}_i)$ it returns the most probable label for $X$ given that $\Omega_S(X) = \Omega_S(\mathbf{x}_i)$.*

The proof of Lemma 1 is presented in appendix C.1.

**Corollary 1.** *The prediction accuracy of an optimal classifier for $\Omega_S(\mathbf{x}_i)$ is equal to the predictive power of set $S$ for $\mathbf{x}_i$, i.e., $\Phi_S(\mathbf{x}_i)$.*

## 4.4 Desired Properties

We now present formalizations of three properties that involve privacy and fairness of classifiers. The properties that we define in this section depend on both the input features used, and the predictions made by a classifier. For a dataset $\mathcal{D}$, we use $S_i \subset F$ to denote the set of features used by a particular classifier to predict the label of data point $\mathbf{x}_i \in \mathcal{D}_\mathcal{X}$. Note that the following properties can be validated for any arbitrary choice of $S_i$ for each data point.

We emphasize that here we are not concerned about how $S_i$ is selected for a given data point and a particular classifier, but we are rather concerned with the social properties of using $S_i$ compared to other feature sets $S'_i \subset F$. (see footnote 2.)

### 4.4.1 Output Property: Fair Prediction Accuracy

In order to define a measure for the fairness in the outputs of a classifier, we use the accuracy equality notion (Verma and Rubin, 2018), and extend it to the individual level as has been suggested by Speicher et al. (2018).

For a classifier $\hat{Y}$ and a dataset $\mathcal{D}$, let $S_i$ be the set of features used to predict the label of $\mathbf{x}_i$. $\hat{Y}$ satisfies fair prediction accuracy if labels of all data points are predicted with equal accuracy, i.e.,

$$\exists \gamma \in (0,1] \ s.t. \ \forall \mathbf{x}_i \in \mathcal{D}_\mathcal{X}, \ acc(\hat{Y}(\Omega_{S_i}(\mathbf{x}_i))) = \gamma \tag{4.2}$$

### 4.4.2 Input Property: Need to Know

The need-to-know property states that for any data point, using any proper subset of the features used by the classifier will decrease the prediction accuracy (i.e., the feature set $S_i$ is minimal with respect to the prediction accuracy):

$$\forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \forall S' \subset S_i,\ acc(\hat{Y}(\Omega_{S'}(\mathbf{x}_i))) < acc(\hat{Y}(\Omega_{S_i}(\mathbf{x}_i))) \tag{4.3}$$

Note that the need-to-know property does not imply that the prediction accuracy must improve monotonically as the number of features that are used by the classifier increases. Furthermore, although we consider accuracy as the criterion for which the use of data is minimized, other measures (e.g., false negative rate) may be more appropriate in specific applications. We leave studying the implications of such alternative definitions of need-to-know for future work.

### 4.4.3 Input Property: Fair Privacy

Fair privacy is determined by the input features used by classifier for each data point. We assume each feature is associated with a non-negative cost that denotes the privacy cost of revealing that feature, and that the privacy cost of each feature is the same across all users. Let vector $\mathbf{c} \in \mathbb{R}^d_{\geq 0}$ denote the privacy costs of the features.

Fair privacy states that the total privacy costs of the used features are equal for all data points, i.e.,

$$\exists \ell \in \mathbb{R}\ s.t.\ \forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}},\ \sum_{f_k \in S_i} \mathbf{c}(k) = \ell \tag{4.4}$$

There are at least two natural cases for the cost vector $\mathbf{c}$:

**Feature Count.** One may choose to treat the privacy costs of all features as equal.

Setting $\mathbf{c} = c.\mathbf{1}_d$ implies that privacy fairness holds when the number of used features is the same for all data points, i.e.,

$$\exists k \in \mathbb{N} \; s.t. \; \forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \; |S_i| = k \tag{4.5}$$

**Feature Match.** Another natural approach is to treat two feature sets as equal-privacy-cost if and only if they contain the same features. This can be formalized by making the total cost of every subset of the features distinct (e.g. $\mathbf{c} = \{2^n | 0 \leq n \leq d-1\}$). In this case, privacy fairness means that the exact same set of features are used to make a prediction for all data points:

$$\exists S \subseteq F \; s.t. \; \forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \; S_i = S \tag{4.6}$$

## 4.5 The Trade-off

In this section we study how the different socially important properties of a classifier defined in Section 4.4 interact. In particular, since all the three properties have important social values, it is natural to ask whether they can be satisfied simultaneously. In other words, we ask whether it is possible for a classifier to use a particular set of input features for each test instance, and satisfy all three properties while maximizing prediction accuracy.

First, we show that there are situations (i.e., datasets) in which an optimal classifier cannot simultaneously satisfy fair privacy, fair accuracy, and need-to-know. Then we present a theorem that precisely characterizes all the datasets in which such a trade-off exists under our definitions. This implies that in general, achieving fairness in the inputs and the outputs of an optimal classifier are incompatible goals.

### 4.5.1 Presenting the Incompatibility

We show that the following proposition is true:

**Table 4.1:** An illustrative dataset.

| | features | | |
|---|---|---|---|
| data point | f1 | f2 | label |
| $\mathbf{x}_1$ | 0 | 0 | - |
| $\mathbf{x}_2$ | 0 | 1 | - |
| $\mathbf{x}_3$ | 0 | 3 | + |
| $\mathbf{x}_4$ | 2 | 3 | - |

**Table 4.2:** Predictive power of each feature set.

| | feature sets | | | |
|---|---|---|---|---|
| data point | $\Phi_\emptyset$ | $\Phi_{\{f1\}}$ | $\Phi_{\{f2\}}$ | $\Phi_{\{f1,f2\}}$ |
| $\mathbf{x}_1$ | 3/4 | 2/3 | 1 | 1 |
| $\mathbf{x}_2$ | 3/4 | 2/3 | 1 | 1 |
| $\mathbf{x}_3$ | 3/4 | 2/3 | 1/2 | 1 |
| $\mathbf{x}_4$ | 3/4 | 1 | 1/2 | 1 |

**Proposition 1.** *When applied to an arbitrary dataset, an optimal classifier cannot be guaranteed to simultaneously satisfy fair privacy, fair accuracy, and need-to-know, unless it is the trivial classifier that does not use any feature values for all data points.*

**Proof.** We provide an example of a dataset for which any optimal classifier can satisfy at most two of fair privacy, fair accuracy, and need-to-know. Table 4.1 presents our example dataset. The dataset contains two features ($f_1$ and $f_2$), and four data points with class labels $y \in \{+, -\}$. Figure 4·1 shows the data points in a 2D plane.

We present our arguments using two different feature cost vectors; each corresponds to one of *feature count* and *feature match* cases introduced in section 4.4.3.

**Feature Count.** Assume that privacy costs of all features are 1, i.e., $\mathbf{c} = \mathbf{1}_2$. Using corollary (1), we know that the prediction accuracy of an optimal classifier for each data point is equal to the predictive power of the selected feature set for that data point. Table 4.2 shows the predictive power of each subset of features for each data point in the dataset.

First, assume an optimal classifier that satisfies fair privacy. Considering the given feature cost vector, the privacy cost for each data point can be either 1 (the classifier uses either $f_1$ or $f_2$) or 2 (the classifier uses both $f_1$ and $f_2$). (Notice that the case of

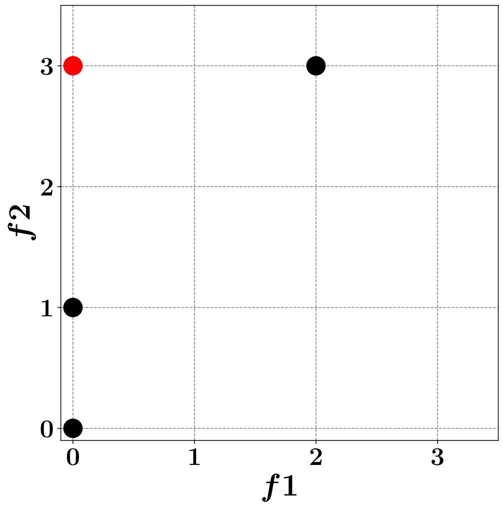


**Figure 4·1:** Example dataset in a 2D plane

using no feature values for all data points is excluded from proposition1.) Therefore, in order to satisfy fair privacy, the privacy cost of all data points should be equal, and is either 1 or 2.

If the privacy cost is 1 for all data points (using either $f_1$ or $f_2$), from Table 4.2 we observe that it is not possible to have equal prediction accuracy for all data points. In particular, the prediction accuracy for $\mathbf{x}_3$ is $\frac{2}{3}$ using $f_1$ and $\frac{1}{2}$ using $f_2$. However, there is no way to have prediction accuracy of $\frac{2}{3}$ for $\mathbf{x}_4$, or $\frac{1}{2}$ for $\mathbf{x}_1$ and $\mathbf{x}_2$ using either $f_1$ or $f_2$. This violates fair prediction accuracy.

If the privacy cost is 2, all the labels can be predicted with accuracy 1.0. However, this violates the need to know property for data points $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_4$ because the same prediction accuracy could be reached using only $f_2$ for $\mathbf{x}_1$ and $\mathbf{x}_2$, or using $f_1$ for $\mathbf{x}_4$.

Therefore, any optimal classifier that satisfies fair privacy when applied on this dataset violates either the fair prediction accuracy or the need to know property. □

**Feature Match.** We could get the same result by assuming feature costs such that the total cost of every feature subset is distinct. In that case, fair privacy reduces to using the same set of features for every data point. From Table 4.2, we observe that the only feature set that satisfies fair accuracy, i.e., the only column with equal predictive power for all data points, is $\{f_1, f_2\}$; and using this set violates need-to-know for $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_4$.

### 4.5.2 Formal Specification

The previous section presents a dataset for which at most two of the properties from Section 4.4 can be satisfied. However, it remains to formalize when precisely a given dataset exhibits the trade-off, which we do in this section. We do this for an optimal classifier under the Feature Match definition of fair privacy (eq.4.6) and we leave generalizing to other definitions of fair privacy for future work. In the common case where privacy costs of the features are unknown, the Feature Match definition— i.e., using the same set of features for all individuals— is a reasonable choice. Similar to Section 4.5.1, in the following we exclude the trivial classifier that does not use any feature values for all data points.

**Theorem 1.** *There exists an optimal non-trivial classifier that satisfies fair privacy, fair accuracy, and need-to-know when applied to a dataset $\mathcal{D}$, if and only if $\mathcal{D}$ satisfies the following condition:*

$$\begin{aligned}\exists \text{ ``non-empty } S\text{''} \subseteq F \ s.t., & \\ & \exists \gamma \in (0,1] \ s.t. \ \forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \ \Phi_S(\mathbf{x}_i) = \gamma \\ & \wedge \\ & \forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \forall S' \subset S, \ \Phi_{S'}(\mathbf{x}_i) < \Phi_S(\mathbf{x}_i)\end{aligned} \tag{4.7}$$

The proof of Theorem 1 is presented in appendix C.2.

Theorem 1 provides a necessary and sufficient condition (eq.4.7) to identify datasets for which an optimal classifier can simultaneously satisfy all the three properties. Notice that this condition is an statement about a dataset and can be verified independently of any classifier. The statement can be written as the following description of a dataset:

*"There is a non-empty feature set that has equal predictive power for all data points in the dataset. Furthermore, all subsets of that feature set have lower predictive power for all points in the dataset."*

Consequently, the negation of (4.7) provides a necessary and sufficient condition for the case where any optimal classifier can satisfy at most two of fair prediction accuracy, fair privacy, and need to know (i.e., datasets in which there is a trade-off between the abovementioned properties of any optimal classifier). By negating (4.7) we find the following characterization of such datasets:

$$\begin{aligned} &\forall \text{ ``non-empty } S\text{''} \subseteq F, \\ &\qquad \exists \mathbf{x}_i, \mathbf{x}_j \in \mathcal{D}_{\mathcal{X}} \ s.t. \ \Phi_S(\mathbf{x}_i) \neq \Phi_S(\mathbf{x}_j) \\ &\qquad \vee \\ &\qquad \exists \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \exists S' \subset S \ s.t. \ \Phi_{S'}(\mathbf{x}_i) \geq \Phi_S(\mathbf{x}_i) \end{aligned} \tag{4.8}$$

For an intuitive interpretation of the above statement, assume an optimal classifier that satisfies fair privacy, i.e., set $S$ is used for all data points in the dataset. Therefore, in order to exhibit the trade-off, using $S$ the classifier should either violate fair prediction accuracy (first clause in (4.8)), or need-to-know (second clause in (4.8)). Thus, showing that for all non-empty $S \subseteq F$ either fair prediction accuracy or need-to-know are violated implies that no optimal non-trivial classifier can satisfy all three properties.

**Corollary 2.** *Given a data set $\mathcal{D}$ and a non-trivial classifier $\hat{Y}$, if $\mathcal{D}$ satisfies (4.8) and $\hat{Y}$ satisfies fair privacy, fair accuracy, and need-to-know when applied to $\mathcal{D}$, then $\hat{Y}$ is not optimal.*

## 4.6 The Trade-off in Real Data

Given the results in the previous section, it is worthwhile to ask whether this trade-off is typical – does it occur often in real-world data? We first develop a practical approach to answering this question for a given dataset, and then we apply our approach to various datasets from the standard UCI machine learning repository Dua and Graff (2017a).

**Algorithm 2:** Verify if a given dataset holds the trade-off.

```
Input: Dataset $\mathcal{D}$ with feature set $F$
Output: Yes/No
initialize queue Q
C = [ ]
Q.put({})
for $f$ in $F$ do
    if $f$ has identical value over all data points then
        remove f from F
while $Q$ is not empty do
    S = Q.get()
    if $S \neq \emptyset$ then
        C.append(S)
    compute $\Phi_S(\mathbf{x}_i)$ for all $\mathbf{x}_i \in \mathcal{D}$
    if $\Phi_S(\mathbf{x}_i) \neq 1$ for all $\mathbf{x}_i \in \mathcal{D}$ then
        for all features $f$ in $F$ whose index is larger than the largest index in $S$ do
            Q.put(S $\cup\{f\}$)
for candidate $S$ in $C$ do
    if $S$ satisfies the 1st clause in (4.8) then
        continue
    if $S$ satisfies the 2nd clause in (4.8) then
        continue
    else
        return No
return Yes
```

### 4.6.1 A Verification Algorithm

For any given dataset, we may apply (4.8) to test it, since (4.8) is a predicate that identifies all and only those datasets for which the trade-off is present. A naive approach to evaluating (4.8) consists of computing $\Phi_{S_k}(\mathbf{x}_i)$ for all subsets $S_k \subseteq F$ and all $\mathbf{x}_i \in \mathcal{D}$. If for each $S_k$ at least one of the two clauses in (4.8) are satisfied, no optimal classifier can simultaneously satisfy fair accuracy, fair privacy, and need-to-know when applied on $\mathcal{D}$. However, the universal quantifier in (4.8) implies a search over the exponential number of subsets in the power set of $F$. Hence, we must consider how to efficiently verify that a given dataset satisfies (4.8). In this section, we introduce a verification algorithm that exploits several structures in the feature susbsets to prune the search space and is efficient in practice.

The pseudocode of our dataset verification algorithm is provided in algorithm 2. The algorithm first generates feature subsets (candidates) for which an optimal classifier could possibly satisfy both fair accuracy and need-to-know. Then it eliminates each candidate that satisfies at least one of the two clauses in (4.8). The algorithm uses an incremental method to generate candidates (i.e., larger sets are generated by adding more features to each of the existing candidates.) This allows the algorithm to recognize many of the candidates that will satisfy (4.8) before actually generating them. This is a key tool for pruning the search space and obtaining a practical algorithm.

The first pruning step is to notice that if a feature $f_i$ has identical values for all data points, removing $f_i$ from a feature subset $S$ does not change the predictive power of that feature subset. That is, $\Phi_S(\mathbf{x}_i) = \Phi_{S \setminus f_i}(\mathbf{x}_i)$ for all $\mathbf{x}_i \in \mathcal{D}$. Therefore, any feature subset that contains $f_i$ violates need-to-know. Consequently, we do not use such features in our candidate generation procedure (lines 4-6).

The second pruning step is to notice that if $\Phi_{S_k}(\mathbf{x}_i) = 1$ for some $S_k \subseteq F$ and some $\mathbf{x}_i$, then any superset $S_k^*$ of $S_k$ violates need to know property (i.e., second clause of (4.8)). This is because predictive power cannot be larger than 1. Therefore, we can prune from our search space all the supersets of any feature set whose predictive power is 1 for at least one data point. As we generate new subsets, we compute the predictive power of each subset for all data points, and we stop adding more features to that subset once a data point is distinguishable in the dataset using that subset (lines 7-14).

Finally, for each generated feature subset (candidate) we first verify the first clause of (4.8); if it is not satisfied we verify the second clause (lines 15-21). Notice that the time complexity of verifying the first clause is linear in the size of the feature subset while the complexity of verifying the second clause is exponential in the size of the feature subset. If all the candidates satisfy at least one of the two clauses in (4.8), we conclude that the given dataset holds the trade-off, i.e., an optimal classifier can satisfy at most two of fair accuracy, fair privacy, and need-to-know for the given dataset.

### 4.6.2 Verifying Real Data

Using Algorithm 2, we find that it is possible to test reasonable-sized datasets and determine whether they exhibit the trade-off introduced in Section 4.5. We obtain 18 datasets which have discrete feature domains from the UCI machine learning repository Dua and Graff (2017a), and apply our verification algorithm to check if the trade-off exists in each dataset. Table D.1 in appendix D summarizes the datasets and the performance of the verification algorithm for each dataset.

We observe that the size of the largest generated candidate for most of the datasets is significantly smaller than the number of features in that dataset, which shows that the superset pruning procedure is effective. The verification algorithm terminates in less than a minute for all cases even though a complete search over the power set of the features would be infeasible in most cases. Also notice that except in one case (Nursery dataset), only verifying the first clause of (4.8) is enough for all data points.

Our algorithm verifies that every dataset we examined exhibits the trade-off between the three properties— hinting that the trade-off is prevalent in real-world data with discrete feature domains.

## 4.7 Summary and Concluding Remarks

In this chapter we argue that the fairness notions for algorithmic decision-making systems should expand to incorporate the inputs (i.e., features) used by a system, and we formulate two of such input properties: *fair privacy* and *need-to-know*.

We prove that in general an *optimal* non-trivial classifier cannot satisfy all of fair privacy, need-to-know, and fair accuracy. Furthermore, we characterize all the datasets in which the above trade-off exists using logical predicates. Finally, we provide an algorithm that exploits several computational efficiencies to verify if the trade-off is present in a given dataset.

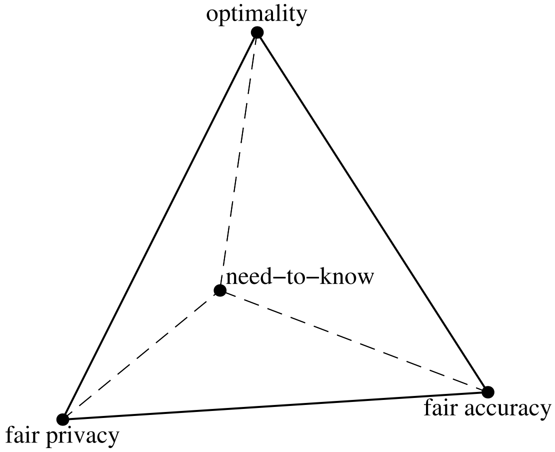


**Figure 4·2:** Summary of the trade-offs

The tetrahedron in Figure 4·2 can be used to summarize our results. In particular, in general the properties at all four vertices cannot be satisfied together. Moreover, each vertex offers a potentially interesting direction for future exploration.

First, if one sets aside optimality to achieve the three socially desirable properties, the question arises then how close to optimal can the performance of such a fair-input and and fair-output classifier be on a given dataset[6].

Second, if one instead sets aside fair privacy, one may seek to achieve the other goals, perhaps following in the general style taken by Noriega-Campero et al. (2019), i.e., using different input features from different individuals.

Third, one may rather choose to set aside need-to-know. For example, Canetti et al. (2019) equalize false positive, false negative, false discovery, and false omission rates across the protected groups by deferring on some decisions (i.e., avoid making a decision for some individuals). However, deferred decisions violate our need-to-know principle which requires the system to use only data inputs that are necessary for improving its predictions.

Finally, one may set aside fair accuracy, perhaps in favor of weaker conditions such as fair mistreatment (Zafar et al., 2017c; Canetti et al., 2019). In that case, the question remains open whether other properties are achievable.

[6] An example of such non-optimal classifier is provided in appendix E.

# Chapter 5

# Auditing Black-Box Prediction Models for Data Minimization Compliance

## 5.1 Introduction

Concerns about the widespread use of data-driven prediction models and their growing reliance on personal data of individuals, have led to a number of data protection laws and regulations in recent years (Munoz et al., 2016; Regulation, 2016; Voigt and Von dem Bussche, 2017). Prominent amongst such regulations is the GDPR (Regulation, 2016) that proposes the *data minimization* principle to control the extent to which personal data can be acquired and used by prediction models. Data minimization is an example of privacy by design approach and it is defined in the GDPR as:

> *Personal data shall be adequate, relevant and limited to what is necessary for the purposes for which they are processed.* (article 5.1.c)

Despite considerable public debate about GDPR and the data minimization principle, to date, only a few works have attempted to *operationalize* (i.e., formally interpret) the legal principle in prediction models. These early works have focused on finding operationalizations of data minimization that tie the purpose of data processing to some performance metric such as prediction accuracy (e.g. in recommender systems (Biega et al., 2020)). Specifically, these works study whether a prediction model can be redesigned to achieve similar prediction performance, while using fewer input features. In the process, these works assume transparent prediction models, where the training data and desired performance metrics are available to the auditor testing the prediction model for compliance to data minimization principle.

In the previous chapter we proposed an operationalization for the data minimization principle that ties the purpose of data processing to classification accuracy. Furthermore, our analysis in Chapter 4 was based on studying the feasibility of data minimization under a transparent model assumption. In other words, given the full knowledge of the prediction algorithm, we studied whether the input data can be reduced while maintaining the model performance measured by classification accuracy.

When operationalizing data minimization, an important case is when an external third-party auditor who does not have access to the prediction model internals would like to audit the system for data minimization compliance. Given that prediction algorithms are often an important business asset, the transparent model assumption is not realistic in this scenario and thus it is reasonable to consider auditing mechanisms in a black-box setting.

In this chapter, we propose an operational definition of the data minimization principle that allows auditing black-box prediction models for compliance at deployment time. In our setting, the auditor does not have access to how the prediction model works or the training data or information about the purpose of the prediction model. All an auditor can do is to query the given black-box prediction model using data points, with a fixed set of input features, and observe the outcomes. We believe our black-box model setting covers many real-world scenarios, as it only requires the designers of prediction models to allow regulators to query their models with prediction instances, but not reveal anything more.

Our key insight is that the auditor can test whether an input feature is needed by the prediction model, by imputing (i.e., guessing) its value and checking the extent to which the outcomes change (i.e., are unstable) for different prediction instances. Intuitively, if the actual value of an input feature is not needed (i.e., can be replaced with a constant) to arrive at similar (stable) outcomes for most prediction instances, then the use of the feature violates the data minimization principle. Note that our instability-based operatlonalization does not require knowledge of the purpose of the prediction model and is independent of performance metrics. Such an operational definition is particularly important given that it is common for companies who provide personalized services to justify data collection simply as "for improving service" (Finck and Biega, 2021).

We show how simple imputations, where the actual values of individual features are replaced with (or assigned) a constant value, can be leveraged as a strategy for limiting data inputs to a prediction system at deployment time. We define a data minimization guarantee that is based on a metric of model instability under feasible simple imputations. While this guarantee induces a procedure for auditing data minimization assuming a finite sample model, we extend the applicability of our auditing framework in two ways. First, we introduce a probabilistic audit that allows the auditor to provide a data minimization guarantee at a fixed confidence level with respect to an underlying data distribution. We adopt a Bayesian approach to provide such a probabilistic guarantee. Second, we address the auditing problem under a constraint on the number of queries to the prediction system and we design auditing algorithms that use the query budget strategically in order to provide a data minimization guarantee.

We cast the problem of allocating a query budget to feasible simple imputations into a multi-armed bandit framework, and we formulate two bandit problems that correspond to different probabilistic audits for some fixed confidence level: a decision problem given a data minimization level, and a measurement problem given a fixed query budget. We provide auditing algorithms for the above problems that use some exploration strategy for investigating the arms. Furthermore, we propose efficient exploration strategies: two inspired by Thompson sampling for Bernoulli bandits, and two that are custom for our setting.

Finally, we use three real-world prediction systems to study the effectiveness of our auditing algorithms. Our experiments show that our auditing algorithms significantly outperform simpler benchmarks in both measurement and decision problems.

## 5.2 Related Work

**Data Minimization** A few papers have addressed the problem of operationalizing data minimization in prediction systems. Biega et al. (2020) propose definitions that are based on accuracy in recommender systems and Galdon Clavell et al. (2020) interpret data minimization as limiting the use of sensitive personal data. For a detailed review of data minimization principle in the literature see Section 2.2

**Model Instability in Machine Learning** The instability of prediction models is studied in both adversarial and non-adversarial settings. The traditional non-adversarial setting is concerned with the instability of a model under different training data (Li and Belford, 2002; Dwyer and Holte, 2007). The adversarial setting studies the effects of data perturbations on the model predictions. Perturbations in *training data* is often called data poisoning and has been studied for various prediction models (Steinhardt et al., 2017). The more relevant problem to our setting is however studying the effect of *test data* perturbations, which is known as evasion attacks at test time (Biggio et al., 2013; Nelson et al., 2010).

Works on evasion attacks generally fall under two categories: computing adversarial instances (i.e., solving the evasion problem) for different models (Kantchelian et al., 2016; Laskov et al., 2014; Szegedy et al., 2013), and certifying the robustness of prediction systems, e.g., in neural networks (Zhang et al., 2018; Weng et al., 2018; Wong et al., 2018; Singh et al., 2018; Gehr et al., 2018; Raghunathan et al., 2018; Wong and Kolter, 2018), or decision trees (Chen et al., 2019).

A key difference between our instability metric and model instability under evasion attacks is in the type of perturbation that we define. While evasion attacks consider perturbations defined as modifying a data instance inside a fixed-radius ball under a particular norm, we consider perturbations defined as projections induced by simple imputations. Moreover, the evasion attack problem concerns the instability of a model prediction locally at each data point whereas the audit problem is concerned with a global notion of instability (see Section 5.3).

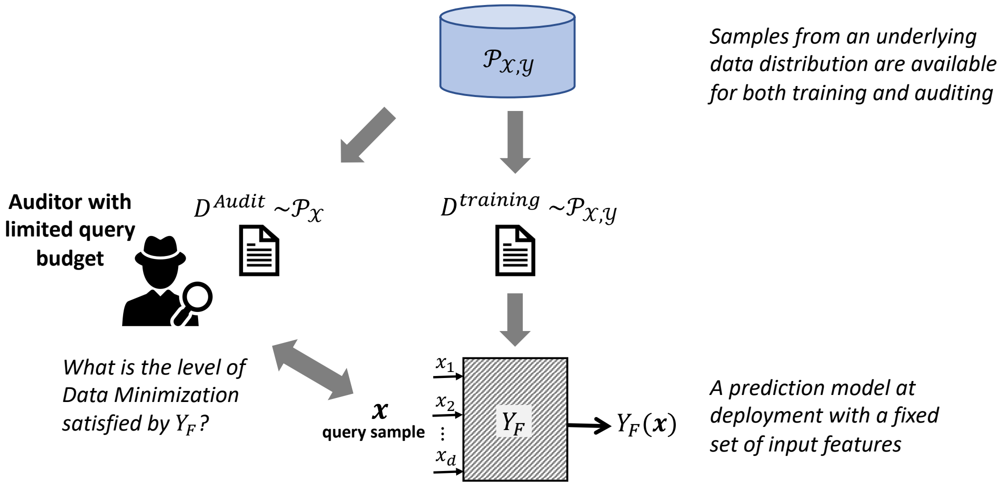
## 5.3 Model Instability-based Data Minimization

In this section we first specify the setting in which we consider the auditing problem for data minimization compliance. Then we explain how simple imputations can be exploited to limit the input features used by a prediction model, and we define a metric for model instability under simple imputations. Finally, we introduce a data minimization guarantee that ensures every input feature is necessary for reaching the model predictions for at least a certain fraction of prediction instances.

### 5.3.1 Notation and Setting

We consider a prediction model being audited at deployment time for adherence to data minimization principle. The model has a fixed set of inputs variables (features) and an output variable from a discrete domain. We assume a black-box setting, i.e., for any prediction instance, an auditor can query the model by specifying the values of all input features and observe the produced output, while the procedure used for generating the output from the inputs is unknown to the auditor. The auditor's goal then is to measure the level of data minimization satisfied by the prediction model using a limited number of queries.

Formally, let $\hat{Y}_F$ denote a prediction model over a set of input features $F = \{f_1, \ldots, f_d\}$, where each feature $f_i$ takes values from domain $\mathcal{X}_i$. $\mathcal{X} = \prod_{i=1}^{d} \mathcal{X}_i$ denotes the input space of $\hat{Y}_F$. For any prediction instance $\mathbf{x} \in \mathcal{X}$, $\hat{Y}_F(\mathbf{x}) \in \mathcal{Y}$ denotes the prediction made by $\hat{Y}_F$ for $\mathbf{x}$, and $\mathcal{Y}$ is the discrete set of targets in the output space. We also consider a distribution $\mathcal{P}_{\mathcal{X}}$ over the input space from which samples (data points) are drawn in deployment[1]. The auditor's challenge is to test whether the the model $\hat{Y}_F$ satisfies data minimization principle over $\mathcal{P}_{\mathcal{X}}$. Figure 5·1 demonstrates our setting for auditing black-box prediction models.


$\mathcal{P}_{\mathcal{X},\mathcal{Y}}$
Auditor with limited query budget
$D^{Audit} \sim \mathcal{P}_{\mathcal{X}}$
$D^{training} \sim \mathcal{P}_{\mathcal{X},\mathcal{Y}}$
$\boldsymbol{x}$
query sample
$x_1$
$x_2$
$\vdots$
$x_d$
$Y_F$
$Y_F(\boldsymbol{x})$

*Samples from an underlying data distribution are available for both training and auditing*

*What is the level of Data Minimization satisfied by $Y_F$?*

*A prediction model at deployment with a fixed set of input features*

**Figure 5·1:** Audit setting.

### 5.3.2 Model Instability under Simple Imputations

**Assessing the need for individual features with simple imputations** To assess the need for individual features, we propose using simple imputations. Simple

[1] We also assume that labeled samples from the joint distribution $\mathcal{P}_{\mathcal{X},\mathcal{Y}}$ were available for training $\hat{Y}_F$; however, we do not make further assumptions about how the training data is used for building the prediction model, i.e., $\hat{Y}_F$ can be any arbitrary function.

imputation is a procedure that is commonly used for handling missing data, and it works by assigning a constant value to a feature $f_i$ in any prediction instance in which $f_i$ is missing. If an auditor finds that imputing a constant value to an input feature $f_i$ has *no or small* effect on outcomes across different prediction instances, then the auditor can conclude that information about the actual values of $f_i$ are *not needed* by $\hat{Y}_F$ to arrive at prediction outcomes. Our idea is to quantify the instability or changes in prediction outcomes to determine the level or extent to which the data minimization principle has been satisfied or violated.

**A metric for model instability under simple imputations** We now define a metric for instability of a prediction model, $\hat{Y}_F$, under simple feature imputations. Formally, we define an imputation function $\tau_{f_i,b}$ such that for any prediction instance $\mathbf{x}$, $\tau_{f_j,b}(\mathbf{x})$ returns a vector in which the value of feature $f_i$ is replaced with $b \in \mathcal{X}_i$. Let $I_{\hat{Y}_F}(\mathbf{x}, f_j, b)$ be a binary indicator variable that is 1 if the prediction made by $\hat{Y}_F$ for $\mathbf{x}$ changes after imputing $f_j$ with $b$, or 0 otherwise. Formally,

$$I_{\hat{Y}_F}(\mathbf{x}, f_j, b) = \begin{cases} 1 & \text{if } \hat{Y}_F\big(\mathbf{x}\big) \neq \hat{Y}_F\big(\tau_{f_j,b}(\mathbf{x})\big) \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

Let $X$ be a random variable that takes on values $\mathbf{x} \in \mathcal{X}$ according to an underlying data distribution $\mathcal{P}_{\mathcal{X}}$. We define the instability of $\hat{Y}_F$ over $\mathcal{P}_{\mathcal{X}}$ with respect to feature $f_j$ and imputation value $b$ as:

$$\beta_j^b = \mathbb{E}_{X \sim \mathcal{P}_{\mathcal{X}}}\big[I_{\hat{Y}_F}(X, f_j, b)\big] \tag{5.2}$$

In other words, $\beta_j^b$ denotes the probability that the prediction for a data point drawn randomly from $\mathcal{P}_{\mathcal{X}}$ changes after imputing $f_j$ by $b$.

### 5.3.3 Instability-based Data Minimization Guarantee

We now define a data minimization guarantee an auditor can offer using our model instability metric $\beta_j^b$. Intuitively, the imputation $b$ that induces the minimum prediction instability for a feature $f_j$ determines how necessary the feature $f_j$ is for generating the predicted outcomes. So a natural data minimization guarantee can be arrived at by finding the greatest lower bound on the need for each individual feature $f_j$ (which in turn requires finding a feasible imputation for each $f_j$ that induces minimum instability).[2] Formally we define:

**Definition 1.** *A prediction model $\hat{Y}_F$ satisfies data minimization at level $\beta$ if there does not exist any feature $f_j \in F$ and any imputation value $b \in \mathcal{X}_j$ such that $\beta_j^b < \beta$. The highest level $\beta$ at which data minimization is satisfied constitutes the best data minimization guarantee an auditor can offer for $\hat{Y}_F$.*

Intuitively, a data minimization guarantee at level $\beta$ ensures that every input feature used by a prediction model is indeed necessary to reach the predictions made for at least a certain fraction, $\beta$, of prediction instances. Note that if a prediction model satisfies data minimization at level $\beta$, it also satisfies data minimization at any level $\beta' < \beta$. In practice, the auditor would be interested in finding the best guarantee, i.e., the largest value of $\beta$ at which a prediction system satisfies data minimization.

## 5.4 Audit Mechanisms for Data Minimization Guarantees

We now consider the challenge of designing efficient black-box audit mechanisms for producing the data minimization guarantee discussed in the previous section. Notice that we defined model instabilities as the expected value of an indicator random variable over an underlying data distribution $\mathcal{P}_{\mathcal{X}}$. In practice, however, the auditor can only query the black-box model with a finite number of samples drawn from the

[2] We only consider the necessity of each individual feature for achieving system outputs, and leave the more general case of considering all feature combinations for future work.

distribution to estimate the model instability. We call this set of available query samples (prediction instances) the audit dataset $\mathcal{D}^{Audit}$.

### 5.4.1 Population Audit

Given an audit dataset and assuming a finite sample model, the model instability with respect to each simple imputation, i.e., each feasible (feature, imputation value) pair, can be computed using the the population mean over all prediction instances in the dataset. In particular, the model instability with respect to feature $f_j \in F$ and imputation value $b \in \mathcal{X}_j$ is:

$$\hat{\beta}_j^b = \frac{1}{|\mathcal{D}^{Audit}|} \sum_{\mathbf{x} \in \mathcal{D}^{Audit}} \left[ I_{\hat{Y}_F}(\mathbf{x}, f_j, b) \right] \tag{5.3}$$

The auditor can then measure the data minimization level by exhaustively searching over all features and imputation values and finding the minimum instability based on the the population mean over all prediction instances in the audit dataset. We call this procedure the *"population audit"*.

However, in many practical auditing scenarios, population audit may not be feasible because the number of queries an auditor can issue to the prediction system is limited and an exhaustive search would exceed the query budget many times over. Furthermore, auditors are often interested in a guarantee that is valid over yet unseen samples drawn from the underlying data distribution. In the following, we address these concerns by defining a probabilistic data minimization guarantee, and introducing the notion of a probabilistic audit which we provide using a Bayesian approach.

### 5.4.2 Probabilistic Data Minimization Guarantee

We address the shortcomings of population audit by introducing a probabilistic audit. In a probabilistic audit, the auditor investigates the black-box prediction model's instability by observing its outputs for a limited number of query data points, and offers a probabilistic guarantee about the data minimization level satisfied by the model. This guarantee allows the auditor to extend the applicability of our instability-based data minimization metric from a finite sample model to a distributional setting.

In particular, analogous to the data minimization guarantee in Definition 1, we define the probabilistic data minimization guarantee as follows:

**Definition 2.** *A prediction model $\hat{Y}_F$ satisfies data minimization at level $\beta$ with $\alpha$ percent confidence if the probability that $\beta_j^b < \beta$ for at least one feature $f_j \in F$ and one imputation value $b \in \mathcal{X}_j$ is less than or equal to $1 - \alpha$.*

Intuitively, satisfying this guarantee at a high confidence $\alpha$ means that with high probability, there does not exist a simple feature imputation using which the model prediction changes for less than $\beta$ fraction of samples drawn from distribution $\mathcal{P}_{\mathcal{X}}$.

### 5.4.3 Probabilistic Audit

The probabilistic auditor can adopt a Bayesian approach to measure the uncertainty about the model instability under different simple imputations. Assuming a prior distribution for each $\beta_j^b$, the success probability of the Bernoulli variable $I_{\hat{Y}_F}(X, f_j, b)$, the auditor can apply the Bayesian update rule to achieve its posterior distribution based on the observations about the model instability under $\tau_{f_j,b}()$. Each observation corresponds to querying the prediction model for investigating the model instability under this imputation for a data point drawn randomly from $\mathcal{P}_{\mathcal{X}}$.

In particular, let $S_j^b$ denote the number of observations for which the simple imputation $\tau_{f_j,b}()$ changes the model prediction, and $F_j^b$ be the number of observations whose prediction does not change. Using the standard choice of modeling the mean of a Bernoulli variable with the Beta distribution and the resulting update rule, we get the posterior distribution $\beta_j^b \sim Beta(a + S_j^b, c + F_j^b)$ when the prior belief is $\beta_j^b \sim Beta(a, c)$ for each feature $f_j$ and imputation value $b \in \mathcal{X}_j$.

Next, we explain how an auditor can use the the posterior distributions of all

$\beta_j^b$s to infer a probabilistic data minimization guarantee, i.e., verify whether with high probability $\alpha$, all $\beta_j^b$s are greater than some level $\beta$. To satisfy the probabilistic data minimization guarantee at level $\beta$ with confidence $\alpha$, Definition 2 requires the probability of the following event to be small (less than or equal to $1-\alpha$): *"for at least one feature $f_j \in F$ and one imputation value $b \in \mathcal{X}_j$, it is true that $\beta_j^b \leq \beta$."* Formally, we can rewrite the statement as:

$$Pr[\exists (f_j \in F, b \in \mathcal{X}_j) \text{ s.t. } \beta_j^b \leq \beta] \leq 1-\alpha \tag{5.4}$$

We can use Boole's inequality to find an upper bound for the the probability in the left hand side of (5.4). We can find a lower bound for the same using the observation that the probability that $\beta_j^b \leq \beta$ for at least one feature $f_j$ and imputation $b$ is greater or equal than the probability that $\beta_j^b \leq \beta$ for any arbitrary $f_j$ and $b$. In particular we have:

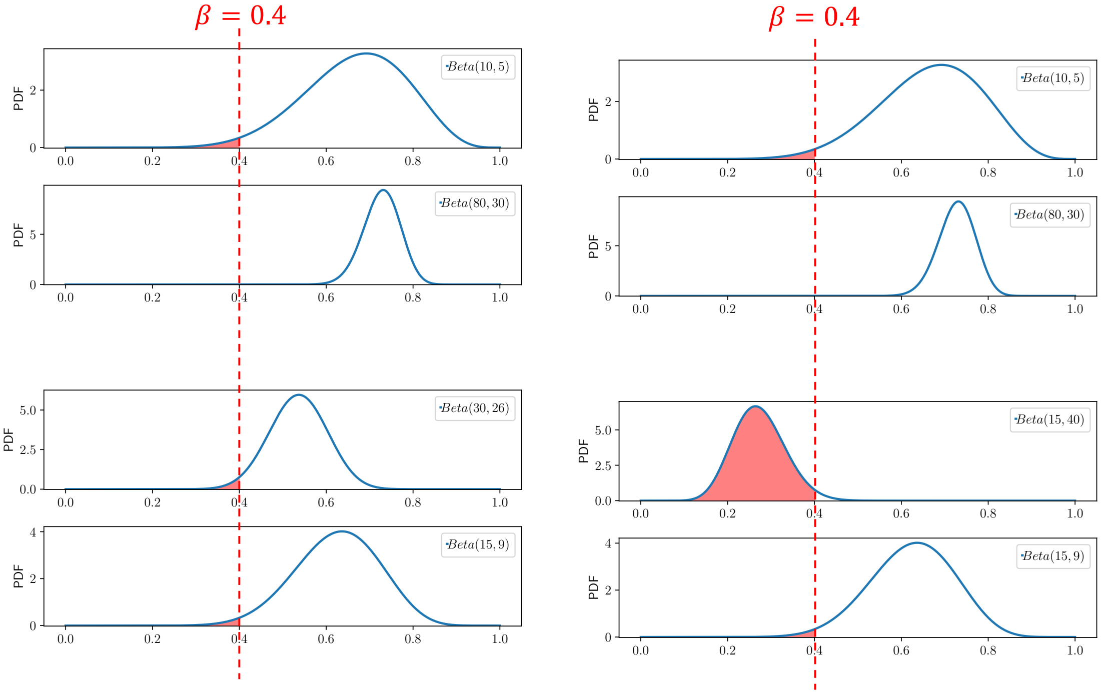
$$\max_{f_j \in F, b \in \mathcal{X}_j} Pr[\beta_j^b \leq \beta] \leq Pr[\exists (f_j \in F, b \in \mathcal{X}_j) \text{ s.t. } \beta_j^b \leq \beta] \leq \sum_{f_j \in F, b \in \mathcal{X}_j} Pr[\beta_j^b \leq \beta] \tag{5.5}$$

Given the posterior distribution of $\beta_j^b$, let $L_j^b(\beta) = F_{Beta}(\beta; a + S_j^b, c + F_j^b)$ denote its cumulative distribution function. We can rewrite (5.5) as:

$$\max_{f_j \in F, b \in \mathcal{X}_j} L_j^b(\beta) \leq Pr[\exists (f_j \in F, b \in \mathcal{X}_j) \text{ s.t. } \beta_j^b \leq \beta] \leq \sum_{f_j \in F, b \in \mathcal{X}_j} L_j^b(\beta) \tag{5.6}$$

The auditor can use the bounds to decide whether or not a prediction system satisfies data minimization at *a given level $\beta$ and a given confidence level $\alpha$* as follows:

If the upper bound in (5.6) is less than or equal to $(1-\alpha)$, then the auditor can declare that prediction system satisfies the probabilistic data minimization guarantee (as the probability that at least one $\beta_j^b$ is less than $\beta$ is smaller than $(1-\alpha)$). Figure 5·2a demonstrates an example of posterior distributions that correspond to this



**(a)** Data minimization is satisfied.

**(b)** Data minimization is not satisfied.

**Figure 5·2:** Example posterior distributions of model instability with respect to different imputations, and a probabilistic data minimization audit at level 0.4 with 90% confidence.

situation.

On the other hand, if the lower bound in (5.6) is greater than or equal to $\alpha$, the auditor can declare that with confidence $\alpha$ the prediction system does not satisfy the probabilistic data minimization guarantee at level $\beta$ since with confidence $\alpha$ there exists at least one simple imputation under which the model-instability is less than $\beta$. Figure 5·2b shows an example of this situation.

If neither of the above conditions apply, the auditor cannot make a decision. That is, it cannot accept or reject data minimization at level $\beta$ with confidence $\alpha$. In this case, the auditor would need to issue more queries to the black-box model and improve its estimate of posterior distributions of $\beta_j^b$s . Figure 5·3 demonstrates an example of posterior distributions that correspond to this situation.

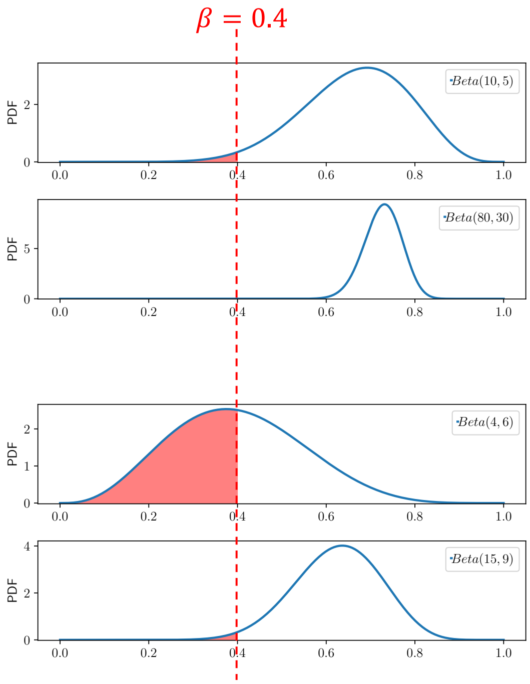


**Figure 5·3:** An example of a situation where a decision cannot be made based on the posterior distributions.

Alternately, if the the auditor is concerned with measuring the best data minimization level that the prediction system satisfies with given confidence $\alpha$, they can leverage the observation that the upper bound in (5.6) is a monotonically increasing function of $\beta$, and apply a binary search to find $\beta$ such that the upper bound in (5.6) is equal to $1-\alpha$. The resulting $\beta$ would be the best data minimization level that can be guaranteed with confidence $\alpha$ based on the posterior distributions.

## 5.5 Auditing With a Limited Query Budget

Using the framework introduced in Section 5.4, an auditor can provide a probabilistic data minimization guarantee based on the posterior distributions that are created for the model instability with respect to each simple imputation. As an auditor investigates the model instability with respect to different imputations, each posterior distribution is updated based on the queries that explore the model instability for that imputation. While a probabilistic guarantee can be provided using the procedure introduced in Section 5.4.3 based on any arbitrary set of observed queries, clearly reducing the uncertainty about the instabilities of certain imputations may be more effective than others in finding a data minimization guarantee with high confidence.

Since it is desirable to limit or minimize the total number of queries used, a challenging problem faced by the auditor is how to distribute queries to investigate the effect of different simple imputations on the model instability. In particular, while a naive auditing approach may allocate equal numbers of queries for updating each posterior distribution, reducing the uncertainty about the model instability with respect to certain imputations might be more helpful than others, when providing a probabilistic data minimization guarantee. Thus an intelligent auditing strategy would spend more queries on investigating those imputations.

We address the above query allocation problem by introducing auditing algorithms that query the prediction system strategically. Hence we cast the problem of allocating a query budget to simple imputations into a bandit framework. Within this framework we consider two stopping criteria, corresponding to the two auditing task types. In particular, we formally define two bandit problems that correspond to the following tasks: (i) measuring the greatest data minimization level satisfied by a prediction model given a fixed query budget, and (ii) deciding whether or not data minimization is satisfied at a given level using the minimum number of system queries. In Section 5.6 we introduce auditing algorithms for each of the above problems.

### 5.5.1 A Multi-armed Bandit Framework

The multi-armed bandit problem (Lattimore and Szepesvári, 2020) is a standard framework for modeling sequential decision problems under uncertainty, in which the actions (choices) are defined by a set of arms. A player sequentially chooses arms to play, and observes noisy signals of their quality, also known as rewards. The goal is then to optimize some utility while acquiring new knowledge about the arms. The query allocation problem for an auditor can be appropriately formulated as a stochastic bandit in which the rewards are modeled by a Bernoulli distribution associated with each arm.

In particular, we consider an arm for each pair $(f_j, b)$ of input feature $f_j \in F$ and feasible imputation value $b \in \mathcal{X}_j$. In each round, the auditor chooses an arm $(f_j, b)$ and observes a binary reward that is a sample from a Bernoulli distribution with success probability $\beta_j^b$. More specifically, each observation of arm $(f_j, b)$ corresponds to evaluating $I_{\hat{Y}_F}(X, f_j, b)$ at a data point $\mathbf{x}$ drawn randomly from $\mathcal{P}_{\mathcal{X}}$. This evaluation requires querying the prediction system to check whether the model prediction for $\mathbf{x}$ changes using the simple imputation associated with $(f_j, b)$. The success probabilities, i.e., the mean rewards of the arms, are unknown to the auditor. Furthermore, we incorporate the Bayesian assumption introduced in Section 5.4 and model the success probability of each arm using a Beta distribution whose shape parameters depend on the observations from that arm.

Bandit algorithms are traditionally developed for optimizing cumulative reward, a goal that requires both exploration and exploitation. However, our auditing problem is a *pure exploration* bandit since the auditor's objective is to explore arms in order to obtain a good estimation of the mean rewards (i.e., model instabilities). Previous approaches to pure exploration problems have focused on two main objectives: minimizing simple regret (Bubeck et al., 2009), and the best arm identification problem (Audibert et al., 2010). Our objective however is different from these, as we explore arms in order to provide a *probabilistic* data minimization guarantee. As explained in Section 5.4, equation (5.6) can be used to provide two types of guarantees depending on whether the data minimization level $\beta$ is fixed or not. Consequently, in the following we define two pure exploration bandit problems, where each is associated to a different auditing task: a decision problem, and a measurement problem. Both problems use the bandit setting described above, but each has a different set of input parameters and a different stopping condition, which results in a different data minimization guarantee.

### 5.5.2 Decision Problem: Fixed Confidence and Fixed Level.

In this problem, the auditor's goal is to guarantee with a given confidence $\alpha$ that whether or not a prediction system satisfies data minimization at a given level $\beta$. A good auditing strategy for this problem tries to use a small number of system queries to provide this guarantee. Formally, based on equation (5.6) and using the bandit framework introduced in 5.5, the auditor seeks to solve the following problem:

*"Given confidence $\alpha$ and data minimization level $\beta$, iteratively select an arm $(f_j, b)$ to explore, and update the posterior distribution of $\beta_j^b$ based on the observed rewards; such that the resulting posteriors induce $\sum_{\substack{f_j \in F \\ b \in \mathcal{X}_j}} L_j^b(\beta) \leq (1-\alpha)$ or $\alpha \leq \max_{\substack{f_j \in F \\ b \in \mathcal{X}_j}} L_j^b(\beta)$ using the minimum number of observations."*

### 5.5.3 Measurement Problem: Fixed Confidence and Fixed Budget.

Alternatively, in this problem the auditor is given a fixed query budget and the goal is to measure $\beta$, the highest level of data minimization that the prediction system is guaranteed to satisfy, with a given confidence $\alpha$. Note that as explained in Section 5.4, for a fixed confidence $\alpha$ and at any state of posterior beliefs, $\beta$ can be computed using a binary search. A good auditing strategy however uses its query budget to provide the highest level of data minimization guarantee it can. Formally, we define the following bandit problem:

*"Given confidence $\alpha$ and query budget $T$, iteratively select an arm $(f_j, b)$ to explore, and update the posterior distribution of $\beta_j^b$ at each round; such that after $T$ rounds*

*the value $\beta$ that satisfies* $\sum_{f_j \in F, b \in \mathcal{X}_j} L_j^b(\beta) = (1-\alpha)$ *is maximized."*

## 5.6 Auditing Algorithms

In this section we present heuristic algorithms for addressing the auditing problems defined in Section 5.5. Our algorithms provide a framework for meeting the probabilistic guarantees required by the decision and measurement versions of a data minimization audit. Furthermore, we address the need to make audits efficient. The key challenge in making an audit efficient is to intelligently select which queries to present to the system under audit. We present a number of strategies, including novel strategies designed for our probabilistic setting. The efficiency of these algorithms in auditing real-world prediction systems is demonstrated in Section 5.7.

Algorithms 3 and 4 present solution strategies for the two versions of the auditing problem. Each algorithm is assumed to be able to query the prediction model $Y_F$, and to be able to sample an audit dataset $\mathcal{D}$. Algorithm 3 summarizes the auditing procedure for solving the decision problem with confidence $\alpha$ and data minimization level $\beta$, where $\alpha$ and $\beta$ are provided as the inputs to the algorithm. The auditing procedure for solving the measurement problem is presented in Algorithm 4. In this algorithm instead of fixing the data minimization level, a fixed query budget $T$ is given as the input and the algorithm returns the level of data minimization that can be guaranteed to be satisfied with confidence $\alpha$.

Each algorithm has at its core an exploration strategy, i.e., a decision about the next query to present to the system, which we denote `SelectArm()` (reflecting the bandit problem viewpoint) in the pseudocodes. This decision uses the current knowledge about the reward distributions of the arms, and some exploration strategies use a given level $\beta$ as well (exploration strategies are presented in section 5.6.1). As described in the previous section, the choice of which bandit arm to sample corresponds to a choice of $(f_j, b)$ and the sampling itself consists of querying $I_{Y_F}(\mathbf{x}, f_j, b)$ where $\mathbf{x}$ is a random data point from $\mathcal{D}$. The algorithms accumulate knowledge about an arm $(f_j, b)$ by maintaining success and failure counters $S_j^b$ and $F_j^b$.

When solving the decision problem (Algorithm 3), the arms are explored iteratively until a decision can be made based on the posterior distributions of the success rates of arms. That is, until the auditor can accept or reject that the prediction model satisfies data minimization at level $\beta$ with confidence $\alpha$ using Eq.(5.6). For the measurement problem on the other hand (Algorithm 4), the auditing algorithm keeps selecting arms and querying the system until all the query budget is used. The largest data minimization level that can be guaranteed with confidence $\alpha$ is then computed using a binary search as explained in section 5.5.

**Algorithm 3:** Audit procedure (Decision Problem)

```
Input: $Y_F$, $\mathcal{D}$, confidence $\alpha$, level $\beta$
$S_j^b \leftarrow 0,\ F_j^b \leftarrow 0; (\forall f_j \in F\ \forall b \in \mathcal{X}_j)$
repeat
    $(f_j, b) \leftarrow$ SelectArm($\beta$)
    Draw $\mathbf{x}$ from $\mathcal{D}$ uniformly at random
    Query $Y_F$ to evaluate $r = I_{Y_F}(\mathbf{x}, f_j, b)$
    Increment either $S_j^b$ or $F_j^b$ based on $r$
until A decision can be made for $\alpha$ and $\beta$;
return The binary decision made using Eq.(5.6)
```

### 5.6.1 Exploration Strategies

Now we present the strategies for `SelectArm()`, the subroutine that selects which imputation to apply to the query sample used in each iteration of the auditing algorithms. While the bandit setting is a natural one for our problems, the probabilistic nature of our problems does not correspond to any classical bandit problem. In Section 5.5 we specified what an optimal exploration strategy is expected to achieve in

**Algorithm 4:** Audit procedure (Measurement Problem)

```
Input: $Y_F$, $\mathcal{D}$, confidence $\alpha$, budget $T$
$S_j^b \leftarrow 0,\ F_j^b \leftarrow 0; (\forall f_j \in F\ \forall b \in \mathcal{X}_j)$
for $t$ in 1 to $T$ do
    Find $\beta^*$ using a binary search
    $(f_j, b) \leftarrow$ SelectArm($\beta^*$)
    Draw $\mathbf{x}$ from $\mathcal{D}$ uniformly at random
    Query $Y_F$ to evaluate $r = I_{Y_F}(\mathbf{x}, f_j, b)$
    Increment either $S_j^b$ or $F_j^b$ based on $r$
return $\beta^*$
```

each of the two auditing problems. Hence, we propose and evaluate heuristic strategies that are either based on classical approaches or new approaches we design in light of the specific nature of our problems.

In particular, we propose four exploration strategies: the first two algorithms are based on Thompson Sampling (originally designed for maximizing the cumulative reward), while the second two are designed specifically for obtaining a lower bound guarantee on the mean reward. Pseudocode for these algorithms is provided in the appendix.

**Thompson Sampling (TS)** Our first approach is based on Thompson sampling (Russo et al., 2017). Thompson sampling is a heuristic for maximizing the expected reward when choosing actions sequentially under uncertainty, and it is shown to have good performance in practice (Chapelle and Li, 2011). It uses Bayesian modeling and a decision strategy called *probability matching*. In particular, in the Bernoulli bandit setting it chooses arms according to their probability of being optimal (maximizing the expected reward) given the current knowledge of the arms at each iteration. It can be efficiently implemented by sampling the posterior beliefs of the mean rewards of the arms at each iteration, and choosing the arm that corresponds to the maximum sample.

We adapt Thompson sampling to be used in our auditing algorithms as an arm selection procedure. In particular, at each iteration and for each arm $(f_j, b)$ a sample $\theta_j^b$ is drawn from $Beta(x; S_j^b + a, F_j^b + c)$ given the current success and failure counters. The arm that corresponds to the *minimum* sample is then selected to be explored next. The idea behind Thomson sampling is to explore an arm according to the chance that reducing the uncertainty about its mean reward would better help finding a data minimization guarantee, which is a probabilistic lower bound on all success probabilities. Therefore, we choose the arm that corresponds to the minimum sample rather than the maximum sample which is selected in original Thompson sampling.

**Top-Two Thompson Sampling (TTTS)** While Thompson sampling is effective in maximizing the expected cumulative reward, it is not a good strategy if the goal is to identify the optimal arm. That is because TS may select a suboptimal arm at the beginning, and as the uncertainty about that arm is being reduced (its posterior belief becomes dense), the algorithm keeps sampling the suboptimal arm and there is hardly any chance for other arms to be explored. In fact it is known that algorithms achieving small cumulative regret cannot be optimal for the best arm identification problem (Bubeck et al., 2009).

Consequently, modifications to TS are proposed to enforce sampling less explored arms more frequently. Top-Two TS (Russo, 2016) addresses this drawback by randomly choosing between two of the best alternatives at each iteration. In particular, with probability $\gamma$ it returns the arm selected by TS, and with probability $1 - \gamma$ it returns an alternative arm by repeating the sampling procedure of TS until a different arm is selected.

Intuitively, identifying the arm with minimum mean reward with high confidence induces a probabilistic lower bound on all mean rewards. Thus TTTS is a natural strategy for our problem as well. However, notice that exact identification of the minimum arm is not necessary for providing a lower bound guarantee.

**Greedy** The previous two algorithms use the posterior beliefs about the mean rewards of the arms, and select arms with respect to their probability of having the minimum mean reward. However, the data minimization guarantee that we seek depends on the probability mass that is below some threshold $\beta$ in all arms. Based on this observation, we develop two new approaches.

In the first approach, instead of sampling the posterior distributions at each iteration, we first evaluate $L^b_j(\beta)$, the cumulative distribution function at $\beta$, for all arms[3]. Then, using a greedy approach, we select the arm whose posterior beta distribution has the maximum probability mass below $\beta$. Given that our data minimization guarantee for both decision and measurement problems depends on either the total sum or the maximum value over all $L^b_j(\beta)$'s, the greedy selection more closely matches our algorithmic goal.

**Probability Matching Using CDFs (PM)** Our second new algorithm adopts the probability matching strategy of TS, and applies it to the cumulative distribution functions of arm rewards at the given threshold $\beta$. That is, it selects arms in proportion to the amount of probability mass that is below $\beta$ in each reward distribution, thus allowing more arms to be explored compared to Greedy.

### 5.6.2 Convergence and Sampling Complexity

From a high level, the sampling strategies are designed to reduce the uncertainty about the mean rewards such that a probabilistic lower bound on all mean rewards can be achieved. While this bound can be inferred if all posterior beliefs are concentrated enough around the true mean of each arm, the sampling strategies focus on exploring the arms updating whose posterior distribution better helps finding a lower bound and

[3] Notice that $\beta$ is an input to `SelectArm()`.

stop exploring arms whose mean is unlikely to be close to the minimum instability. For this task, a problem instance in which multiple arms have a mean reward close to the minimum instability of all arms is harder to solve. However this problem is easier than the best arm identification problem in the sense the exact identification of the best arm is not required as far as a probabilistic lower bound on all arms can be achieved.

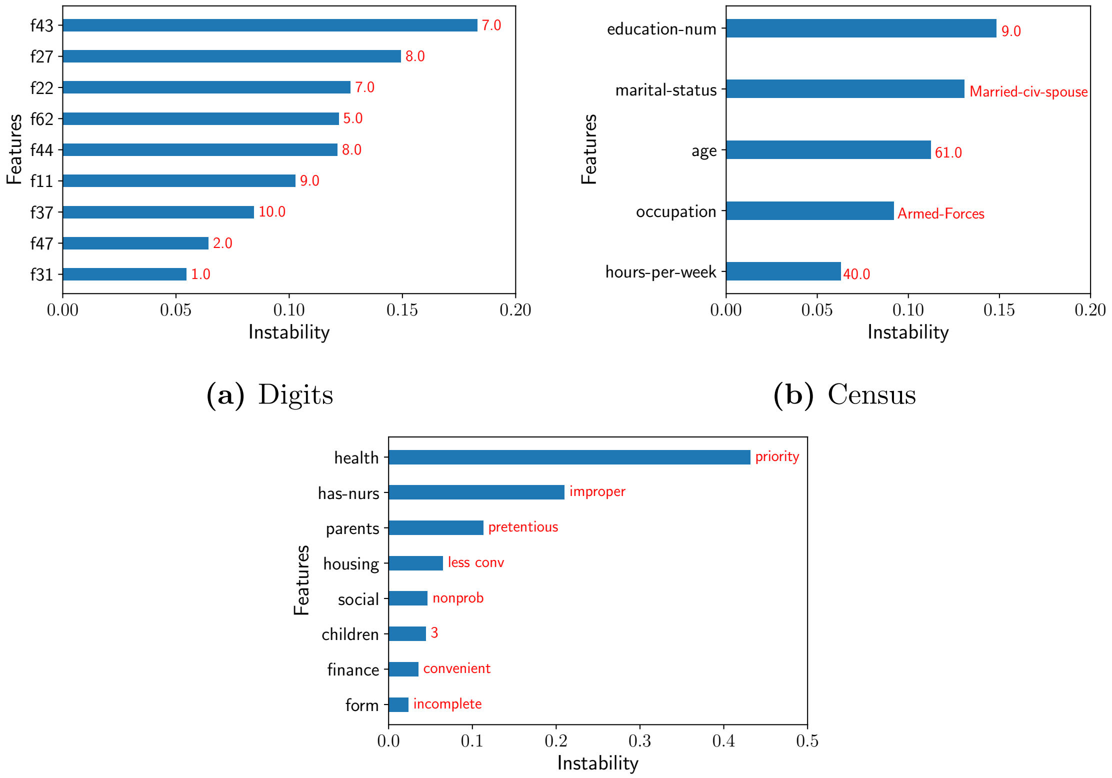
The sample complexity of the best arm identification problem is previously studied by Bubeck et al. (2009), and the convergence rate of TS-based strategies are studied for both the expected cumulative regret (Russo and Van Roy, 2014) and the best arm identification problem (Russo, 2016). In particular it is shown that using Top-Two TS, the probability of selecting a sub-optimal arm converges to zero at an exponential rate (Russo, 2016).

Intuitively, identifying the arm with minimum mean reward with high confidence induces a probabilistic lower bound on all mean rewards. However, the bandit problems introduced in this chapter does not correspond to the classical bandit problems and better convergence rates might be achievable using the exploration strategies in this chapter.

Finally, note that the measurement algorithm (Algorithm 4) is applicable given any query budget, i.e., for any status of posterior beliefs. Therefore, if the decision algorithm cannot make a decision using some given query budget and confidence, Algorithm 4 can be applied to find the largest data minimization level that the auditor can guarantee.

## 5.7 Auditing Real-world Prediction Systems

In this section we study the effectiveness of the algorithms introduced in section 5.6 in auditing real-world prediction systems for data minimization compliance.



**(a)** Digits

**(b)** Census

**(c)** Nursery

**Figure 5·4:** Population audit of three real-world prediction models.

We build prediction systems using datasets from the UCI machine learning repository (Dua and Graff, 2017b). We use three datasets that have discrete features and discrete target variables, and we apply different learning algorithms together with standard feature selection and model validation methods to build a prediction model. In particular, we use the following prediction systems.

**Digits/SVM** A dataset of 3823 images of hand-written digits from the MNIST (LeCun et al., 1998) database is used. Each data point is an $8 \times 8$ matrix whose elements are integers from the range $[0, 16]$, along with a label from the set $\{0, 1, \ldots, 9\}$. A support vector machine with linear kernel is used to build a classifier that predicts the label associated with each image. 50% of data points are used to train the classifier, and a recursive feature elimination procedure is applied to select 9 features.

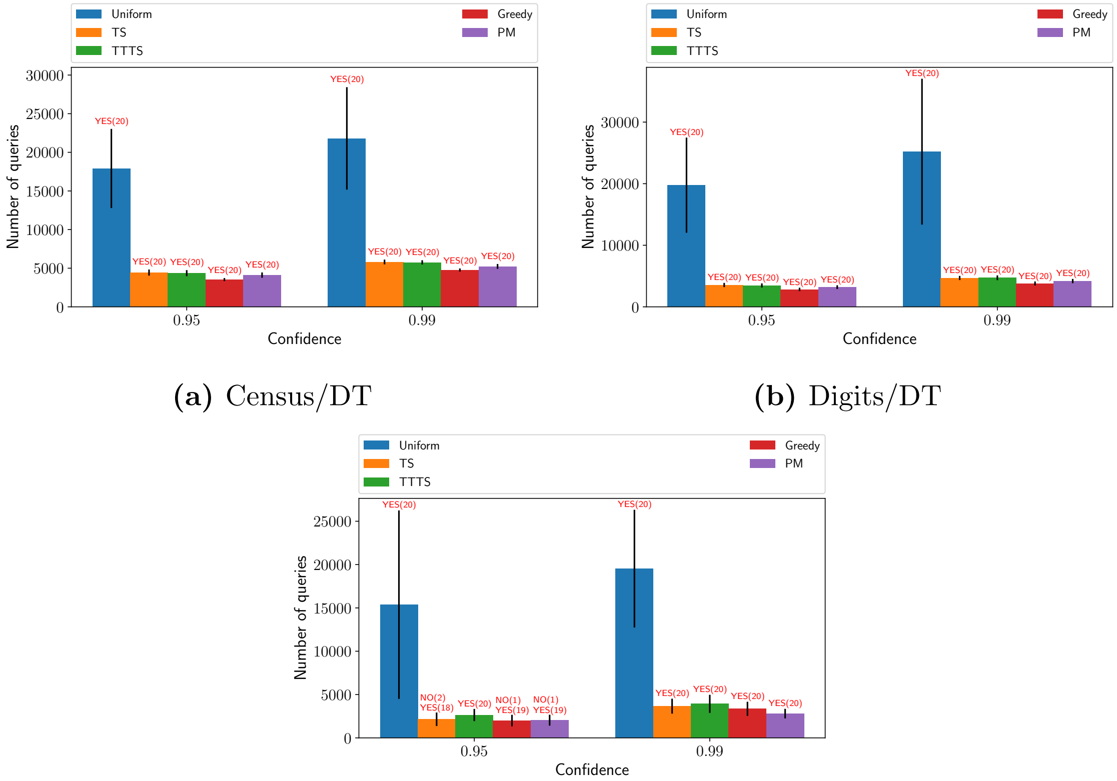
**Census/Decision Tree** A dataset of $\sim 30K$ individuals from the US Census database is used. Eleven features with a discrete domain are available for each individual, and the goal is to predict whether a person makes over $\$50K$ a year. A decision tree is built using 20% of data samples as the training data, and a recursive feature elimination procedure is applied to select 5 features.

**Nursery/Decision Tree** This dataset contains 8 discrete features from $\sim 13K$ applicants for nursery schools in Slovenia. A label from a set of 5 priority groups is assigned to each applicant, and we use 20% of data samples to build a decision tree for predicting the priority group of each applicant.

First, we assume unlimited system queries are allowed and we apply the population audit procedure introduced in Section 5.4.1, which measures data minimization with respect to an audit dataset. The bar charts in figure 5·4 show the computed empirical instabilities with respect to each input feature for the above prediction systems. For each input feature, we show the instability in the case of using the imputation value which results in the lowest instability. The corresponding imputation value is printed on each bar. We observe that in all the three systems, all input features have non-zero instability. And the data minimization level is between 2% to 6% in these systems.

Now we apply our algorithms which are designed for the more realistic scenario of auditing with a limited query budget, and provide a probabilistic guarantee with respect to the underlying data distribution. In the following experiments, for each system we use the whole data as the audit dataset from which query samples are drawn. All prior distributions for $\beta$ parameters are $Beta(1/2, 1/2)$ (i.e., the Jeffreys prior). The decision parameter $\gamma$ in `TTTS` strategy is set to 1/2.

In our first experiment, we apply Algorithm 3 to decide whether each of the above systems satisfy data minimization at 1% level. We perform this task for 0.95 and



**(a)** Census/DT

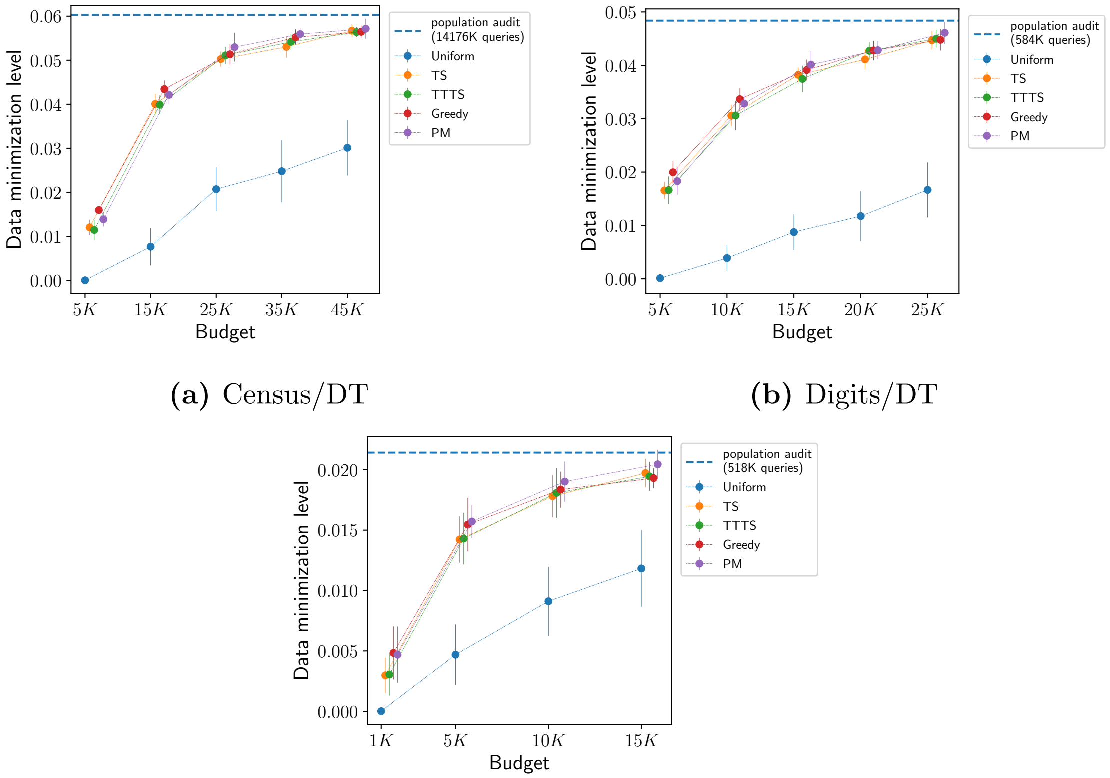
**(b)** Digits/DT

**(c)** Nursery/DT

**Figure 5·5:** Auditing for 1% data minimization.

0.99 confidences, and using the exploration strategies introduced in section 5.6.1. As a baseline algorithm, we also implement a uniform exploration strategy that investigates all (feature, imputation value) pairs uniformly until a decision can be made.

The bar chart in Figure 5·5 shows the average number of system queries used by each exploration algorithm and for each confidence level over 20 runs. The error bars show one standard deviation and the resulting Yes/No decisions are printed over each bar. (the frequencies of decisions are shown in parentheses.) We observe that all three systems satisfy data minimization at 1% level, and our exploration algorithms reach this decision using significantly fewer queries, compared to the uniform exploration method. Furthermore, the algorithms designed specifically for our auditing purpose (`Greedy` and `PM`) are on average slightly more effective than algorithms that are based



**(a)** Census/DT

**(b)** Digits/DT

**(c)** Nursery/DT

**Figure 5·6:** Measuring data minimization with 95% confidence.

on standard Thomson sampling.

Next we apply Algorithm 4 to measure the level of data minimization satisfied by each system using different query budgets. We consider a .095 confidence for this measurement task, and in order to have a comparison point if the were no limit on the query budget, we compute the probabilistic data minimization level using the population audit, i.e., using all the data points in the audit dataset.

Figure 5·6 shows the level of data minimization that each algorithm can guarantee for each system and using different query budgets. In each experiment we increase the budget until one of the algorithms reaches the level measured by population audit. We observe that all of our exploration strategies significantly outperform the uniform exploration, i.e., they can guarantee a higher level of data minimization at

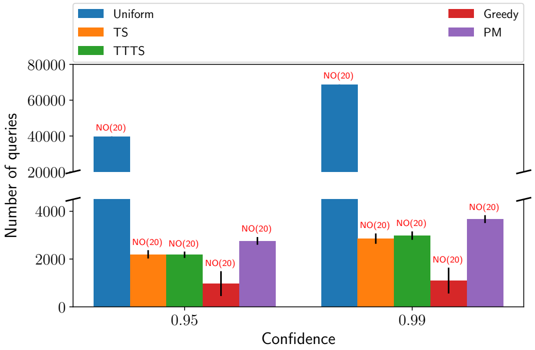


**Figure 5·7:** Auditing a prediction model with excessive inputs

the given confidence for each query budget, and they can reach the level guaranteed by population audit using less than 5% of queries used by population audit. Finally, we observer that for smaller query budgets `Greedy` and `PM` are more effective than standard Thomson sampling algorithms.

Finally, we illustrate the performance of our methods for a hypothetical scenario in which a data minimization audit may be particularly needed. We consider a case in which a prediction system has an input that is not actually used by the prediction model, i.e., the system collects excessive information from its users. We obtain such system by embedding the Census/DT model in another prediction system that asks for an additional attribute of the Census dataset but returns the output of the embedded model.

We apply the decision auditing algorithm for 1% data minimization level to this systems. Figure 5·7 presents the number of queries used, and the output of the auditing algorithms for each exploration strategy and two confidence levels. We observe that all algorithms return "No", i.e., detect that data minimization is not satisfied, as desired. More importantly, the exploration algorithms we propose use an order of magnitude less queries to make this decision. And we particularly note that our `Greedy` strategy is significantly more effective than the other exploration strategies considered.

## 5.8 Summary and Concluding Remarks

In this chapter we provide an operationalization of data minimization principle that is applicable to auditing prediction models. We propose using model instability as the purpose for which data needs to be minimized and we suggest exploiting simple imputations as a tool for limiting data inputs at the test time. Adopting a Bayesian approach and a bandit framework, we provide efficient auditing algorithms that measure model instability with respect to different simple imputations and provide a probabilistic data minimization guarantee.

We initiated the audit framework assuming discrete feature values. In the scenarios where feature domains are relatively large (e.g., integers), more efficient bandit algorithms can be developed using ideas similar to (Yue et al., 2012; Jedor et al., 2019; Kumar et al., 2019), or Wang et al. (2008) for infinitive domains. Other directions for future research include extending our individual features guarantee to the case of all feature combinations, and studying the implications of instability-based data minimization on different fairness notions similar to works that have done it for accuracy-based data minimization (Rastegarpanah et al., 2020).

Also it is interesting to explore the relationship between our model-instability metric for the necessity of features and other feature importance metrics, or more broadly, the feature engineering methods. In our experiments, we had applied standard backward elimination feature selection methods in order to build the example systems. Alternatively, one can study the effect on model instability of other feature selection methods. Furthermore, our model-instability metric itself suggests a feature importance metric that might be useful for feature selection too. Similar instability-based feature importance metrics are previously proposed by Datta et al. (2015).

However, to the best of our knowledge, the distributional instability metric and the idea of using a lower bound on the instabilities with respect to all possible simple imputations are novel.

Finally, we emphasize that our approach in this chapter does not solve the general data minimization problem in its full social context. Rather, we are proposing tools that help address some aspects of the bigger data minimization problem.

# Chapter 6

# Conclusions

The increasing interest in responsible machine learning has resulted in a fast-growing number of publications in recent years. Research in this area covers a wide range of topics, from formalizing notions and trade-offs, to developing responsible algorithms and practical auditing tools. Despite the volume and variety of published results, the field is still at its early stages considering open questions about its conceptual, theoretical, and practical aspects.

The first part of this thesis, Chapter 3, introduced a new technique for improving socially relevant properties of collaborative filtering recommender systems. Previous proposals for this problem are often not easily adapted to different measures, and they generally require the ability to modify either existing system inputs, the recommender algorithm, or the system outputs. Alternatively, this thesis proposed a method that is based on the idea of perturbing the training data distribution by augmenting the system input with additional user-item ratings, which we call antidote data. We took as our model system the matrix factorization approach to recommendation, and we developed optimization-based solutions for computing antidote ratings.

Although we developed the antidote data generation framework for matrix factorization recommender algorithms, the applicability of this approach can be extended beyond factorization-based models using previous proposals for poisoning attacks in different models. Finally, the effectiveness of antidote data framework was demonstrated using a set of measures that are based on existing notions for polarization and fairness of recommender systems. However, our generic framework is applicable for improving any property that can be expressed as a differentiable function of the predicted ratings.

The second part of this thesis, Chapters 4 and 5, focused on operationalizing data minimization, a privacy principle that exists in data protection regulations. Chapter 4 proposed a formalization of data minimization that is based on classification accuracy. We called the resulting property the need-to-know principle, and described how this principle provides a basis for expanding fairness notions to incorporate data inputs used by a decision system. In particular, we introduced another input property, the fair-privacy principle, and proved that in general an optimal classifier cannot simultaneously satisfy need-to-know, and fair-privacy and fair-accuracy. This incompatibility result offers an interesting direction for future research: if one sets aside any of the properties in Chapter 4, are the remaining properties achievable simultaneously? And in what circumstances is satisfying that subset of properties desirable?

Since the results in Chapter 4 are obtained assuming a Bayes optimal classifier that returns the most probable label given any arbitrary subset of feature values, the presented trade-off cannot be avoided regardless of how partially known inputs are handled by a classifier. However, in practice we are usually given a prediction model with a fixed set of input features at the test time and the model internals (e.g., the prediction algorithm) are unknown.

Therefore, in Chapter 5 we addressed the problem of auditing black-box prediction models for compliance with the data minimization principle, and we argue that an operationalization based on model instability is more practical in this setting. Given that the set of input features are fixed, we showed how simple imputations can be leveraged as a strategy for limiting data inputs to a prediction system at deployment time.

In practice, usually the number of system queries available to an auditor is limited. Therefore, we adopted a Bayesian approach and a bandit framework to provide efficient auditing algorithms that measure model instability with respect to different simple imputations and provide a probabilistic data minimization guarantee.

Finally, the auditing tool introduced in Chapter 5 suggests a potentially interesting direction for future research where the auditing problem is viewed from the perspective of the system designer. In particular, a system designers may want to assure that the prediction system will satisfy the instability-based data minimization guarantee introduced in this thesis. Given that the system designer has access to the prediction algorithm, this problem translates into providing provable lower bounds (verification algorithms) on the model instability under simple imputations for particular prediction models.

# Appendix A

# Derivation of the Gradients

## A.1 Derivation of $\frac{\partial \mathbf{V}}{\partial \tilde{\mathbf{X}}}$

Let $E$ be the value of the objective function in (1). Assuming that the factorization algorithm finds a local optimum of $E$, we have $\frac{\partial E}{\partial \mathbf{v}_j} = 0$, which give us the following:

$$\sum_{i \in \Omega_j} (x_{ij} - \mathbf{u}_i^\intercal \mathbf{v}_j)\mathbf{u}_i + \sum_{i=1}^{n'} (\tilde{x}_{ij} - \tilde{\mathbf{u}}_i^\intercal \mathbf{v}_j)\tilde{\mathbf{u}}_i = \lambda \mathbf{v}_j \tag{A.1}$$

From the above equation we can show that the following formula for $\mathbf{v}_j$ holds at a local optimum of $E$:

$$\mathbf{v}_j = (\sum_{i \in \Omega_j} \mathbf{u}_i \mathbf{u}_i^\intercal + \tilde{\mathbf{U}}\tilde{\mathbf{U}}^\intercal + \lambda \mathbf{I}_\ell)^{-1} (\sum_{i \in \Omega_j} x_{ij}\mathbf{u}_i + \sum_{i=1}^{n'} \tilde{x}_{ij}\tilde{\mathbf{u}}_i) \tag{A.2}$$

Therefore, assuming that an infinitesimal change in $\tilde{x}_{ij}$ only results in first order corrections we can write:

$$\frac{\partial \mathbf{v}_j}{\partial \tilde{x}_{ij}} = (\sum_{i \in \Omega_j} \mathbf{u}_i \mathbf{u}_i^\intercal + \tilde{\mathbf{U}}\tilde{\mathbf{U}}^\intercal + \lambda \mathbf{I}_\ell)^{-1} \tilde{\mathbf{u}}_i \tag{A.3}$$

## A.2 Gradients of the objective functions

### A.2.1 Polarization

$$\begin{aligned}
\frac{\partial R_{pol}}{\partial \hat{x}_{ij}} &= \frac{2}{n^2 d} \sum_{k \neq i} (\hat{x}_{ij} - \hat{x}_{kj}) && \text{(A.4)} \\
&= \frac{2}{nd} (\hat{x}_{ij} - \frac{1}{n} \sum_{k=1}^{n} \hat{x}_{kj}) && \text{(A.5)} \\
&= \frac{2}{nd} (\hat{x}_{ij} - \mu_j) && \text{(A.6)}
\end{aligned}$$

where $\mu_j$ is the average estimated rating for item $j$.

### A.2.2 Individual fairness

For $(i, j) \in \Omega$ we have:

$$\begin{aligned}
\frac{\partial R_{indv}}{\partial \hat{x}_{ij}} &= \frac{1}{n^2} (\sum_{k > i} 2(\ell_i - \ell_k) \frac{\partial \ell_i}{\partial \hat{x}_{ij}} + \sum_{k < i} -2(\ell_k - \ell_i) \frac{\partial \ell_i}{\partial \hat{x}_{ij}}) && \text{(A.7)} \\
&= \frac{2}{n^2} \sum_{k \neq i} (\ell_i - \ell_k) \frac{\partial \ell_i}{\partial \hat{x}_{ij}} && \text{(A.8)} \\
&= \frac{4(\hat{x}_{ij} - x_{ij})}{n^2 |\Omega^i|} \sum_{k \neq i} (\ell_i - \ell_k) && \text{(A.9)} \\
&= \frac{4(\hat{x}_{ij} - x_{ij})}{n |\Omega^i|} (\ell_i - \frac{1}{n} \sum_{k=1}^{n} \ell_k) && \text{(A.10)}
\end{aligned}$$

therefore,

$$\frac{\partial R_{indv}}{\partial \hat{x}_{ij}} = \begin{cases} \frac{4(\hat{x}_{ij} - x_{ij})}{n|\Omega^i|} (\ell_i - \mu_{indv}) & (i, j) \in \Omega \\ 0 & (i, j) \notin \Omega \end{cases} \tag{A.11}$$

where $\mu_{indv}$ is the average of user losses.

### A.2.3 Group fairness

Assume $\mathcal{G}()$ is a function that maps each user/item to its group label. For $(i,j) \in \Omega$ we have:

$$\begin{aligned}
\frac{\partial R_{grp}}{\partial \hat{x}_{ij}} &= \frac{2}{g^2} \sum_{k \neq \mathcal{G}(i)} (L_{\mathcal{G}(i)} - L_k) \frac{\partial L_{\mathcal{G}(i)}}{\partial \hat{x}_{ij}} && \text{(A.12)} \\
&= \frac{4(\hat{x}_{ij} - x_{ij})}{g^2 |\Omega_{\mathcal{G}(i)}|} \sum_{k \neq \mathcal{G}(i)} (L_{\mathcal{G}(i)} - L_k) && \text{(A.13)} \\
&= \frac{4(\hat{x}_{ij} - x_{ij})}{g |\Omega_{\mathcal{G}(i)}|} (L_{\mathcal{G}(i)} - \frac{1}{g} \sum_{k=1}^{g} L_k) && \text{(A.14)}
\end{aligned}$$

therefore,

$$\frac{\partial R_{grp}}{\partial \hat{x}_{ij}} = \begin{cases} \frac{4(\hat{x}_{ij} - x_{ij})}{g|\Omega_{\mathcal{G}(i)}|} (L_{\mathcal{G}(i)} - \mu_G) & (i,j) \in \Omega \\ 0 & (i,j) \notin \Omega \end{cases} \tag{A.15}$$

where $\mu_G$ is the average of group losses.

# Appendix B

# Heuristic Algorithms

In this section we present the pseudocode of the heuristic algorithms introduced in section 6.2 for generating antidote data to improve individual and group fairness.

## B.1 Heuristic 1

1. Start from a single antidote user $\tilde{\mathbf{x}}_1^{(0)}$.
2. Compute $\mathbf{U}, \tilde{\mathbf{u}}_1, \mathbf{V} = \boldsymbol{\Theta}(\mathbf{X}; \tilde{\mathbf{x}}_1^{(0)})$.
3. Compute $\frac{\partial R(\hat{\mathbf{X}})}{\partial \tilde{x}_{1j}}$ for each item $j$ using (9).
4. If $\frac{\partial R(\hat{\mathbf{X}})}{\partial \tilde{x}_{1j}} > 0$ set $\tilde{x}_{1j} = \mathbb{M}_{min}$ Else set $\tilde{x}_{1j} = \mathbb{M}_{max}$.
5. Copy $\tilde{\mathbf{x}}_1$ $\alpha$ times to generate $\tilde{\mathbf{X}}$ for a given budget $\alpha$.

## B.2 Heuristic 2

1. Compute $\nabla_{\hat{\mathbf{X}}} R(\hat{\mathbf{X}})$ and reshape it into an $n \times d$ matrix $\mathbf{G}$.
2. Set $\mathbf{d}[j] = \mathbf{g}_j^\intercal \mathbf{U}^\intercal \mathbf{1}_\ell$ for each item $j$ using $\mathbf{G}$ and the original factor $\mathbf{U}$.
3. If $\mathbf{d}[j] > 0$ set $\tilde{x}_{1j} = \mathbb{M}_{min}$ Else set $\tilde{x}_{1j} = \mathbb{M}_{max}$.
4. Copy $\tilde{\mathbf{x}}_1$ $\alpha$ times to generate $\tilde{\mathbf{X}}$ for a given budget $\alpha$.

# Appendix C

# Proofs

## C.1 Lemma 1

We write the right hand side of (4.1) as:

$$\sum_{c \in \mathcal{Y}} Pr[\hat{Y}(\Omega_S(X)) = c, Y(X) = c | \Omega_S(X) = \Omega_S(\mathbf{x}_i)] \tag{C.1}$$

Since $\hat{Y}(\Omega_S(X))$ only depends on the values of the features in $S$, $\hat{Y}(\Omega_S(X))$ and $Y(X)$ are conditionally independent given a set of fixed values $\Omega_S(X) = \Omega_S(\mathbf{x}_i)$. Therefore (C.1) is equal to:

$$\sum_{c \in \mathcal{Y}} Pr[\hat{Y}(\Omega_S(X)) = c | \Omega_S(X) = \Omega_S(\mathbf{x}_i)] Pr[Y(X) = c | \Omega_S(X) = \Omega_S(\mathbf{x}_i)] \tag{C.2}$$

For any $\Omega_S(\mathbf{x}_i)$, let $p_c = Pr[Y(X) = c | \Omega_S(X) = \Omega_S(\mathbf{x}_i)]$ and $p^* = \max_{c \in \mathcal{Y}} p_c$. Also let $\hat{p}_c = Pr[\hat{Y}(\Omega_S(X)) = c | \Omega_S(X) = \Omega_S(\mathbf{x}_i)]$ for an arbitrary classifier $\hat{Y}$. Using (C.2) we can write the following for any classifier $\hat{Y}$:

$$acc(\hat{Y}(\Omega_S(\mathbf{x}_i))) = \sum_{c \in \mathcal{Y}} p_c \hat{p}_c \leq \sum_{c \in \mathcal{Y}} p^* \hat{p}_c = p^*$$

Therefore, $p^*$ is an upper bound for the prediction accuracy of any classifier applied on $\Omega_S(\mathbf{x}_i)$. Now let $c^* = \arg\max_{c \in \mathcal{Y}} p_c$, the prediction accuracy of a classifier that deterministically returns $c^*$ for $\Omega_S(\mathbf{x}_i)$ (i.e., $\hat{p}_c = 0$ for all $c \neq c^*$, and $\hat{p}_{c^*} = 1$) is $p^*$; therefore, such classifier is optimal.

Finally, we show that the prediction accuracy of any classifier with $\hat{p}_{c^*} < 1$ is lower than $p^*$. Assume a classifier $\hat{Y}'$ for which $\hat{p}_{c^*} = 1 - \epsilon$ and $\hat{p}_{c'} = \epsilon$ for some $\epsilon > 0$ and some $c' \in \mathcal{Y}$ such that $p_{c'} < p^*$. Thus,

$$acc(\hat{Y}'(\Omega_S(\mathbf{x}_i))) = p^*(1-\epsilon) + p_{c'}\epsilon = p^* + \epsilon(p_{c'} - p^*) < p^* \quad \square$$

## C.2 Theorem 1

For a classifier $\hat{Y}$ applied to a dataset $\mathcal{D}$, let $S_i$ denote the set of features used from $\mathbf{x}_i$. First we repeat and name the following definitions from section 4.4,

**Fair Accuracy** *(p1)*:

$$\exists \gamma \in (0,1] \ s.t. \ \forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \ acc(\hat{Y}(\Omega_{S_i}(\mathbf{x}_i))) = \gamma$$

**Fair Privacy** *(p2)*:

$$\exists S \subseteq F \ s.t. \ \forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \ S_i = S$$

**Need-To-Know** *(p3)*:

$$\forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \forall S' \subset S_i, \ acc(\hat{Y}(\Omega_{S'}(\mathbf{x}_i))) < acc(\hat{Y}(\Omega_{S_i}(\mathbf{x}_i)))$$

Let $\mathcal{H}_{opt}$ be the set of all optimal non-trivial classifiers for $\mathcal{D}$. Assume there exists an optimal non-trivial classifier that satisfies all of *p1*, *p2*, and *p3* when applied on $\mathcal{D}$, i.e., the following statement is true:

$$\exists \hat{Y} \in \mathcal{H}_{opt} \ s.t. \ p1 \wedge p2 \wedge p3 \tag{C.3}$$

From *p2* we infer that the same set of features is used by the classifier to predict the label of all the data points. We call this set $S$ and we replace $S_i$ with $S$ in *p1* and *p3* ($S$ is non-empty since $\hat{Y}$ is non-trivial). Moreover, since $\hat{Y}$ is an optimal classifier, by Corollary (1) we can replace the prediction accuracy of $\hat{Y}$ for any data point and any feature set with the predictive power of that feature set for that data point. Thus, from (C.3) we infer that the following statement is true:

$$
\begin{aligned}
\exists \text{ ``non-empty } S\text{''} \subseteq F \; s.t., \quad & \\
& \exists \gamma \in (0,1] \; s.t. \; \forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \;\; \Phi_S(\mathbf{x}_i) = \gamma \\
& \wedge \\
& \forall \mathbf{x}_i \in \mathcal{D}_{\mathcal{X}}, \forall S' \subset S, \;\; \Phi_{S'}(\mathbf{x}_i) < \Phi_S(\mathbf{x}_i)
\end{aligned}
\tag{C.4}
$$

On the other hand, assume we are given a dataset for which statement (C.4) is true. We can then define a classifier $\hat{Y}$ such that it uses the features in $S$ for all $\mathbf{x}_i \in \mathcal{D}$, and it returns $\arg\max_{c \in \mathcal{Y}} Pr[Y(X) = c | \Omega_{S_k}(X) = \Omega_{S_k}(\mathbf{x}_i)]$ for any $S_k \subseteq F$ and $\mathbf{x}_i \in \mathcal{D}$, i.e., $\hat{Y}$ is optimal. Therefore, $acc(\hat{Y}(\Omega_{S_k}(\mathbf{x}_i))) = \Phi_{S_k}(\mathbf{x}_i)$ and from (C.4) we infer that $\hat{Y}$ satisfies $p1$ and $p3$. Furthermore, $\hat{Y}$ satisfies $p2$ because it uses $S$ for all data points, and is optimal by definition. Therefore, statement (C.3) is satisfied for $\mathcal{D}$.

Thus the property defined in (C.4) is a necessary and sufficient condition to recognize datasets for which there exists an optimal classifier that satisfies all properties *p1*, *p2*, and *p3*. □

# Appendix D

# Dataset Verification Results

**Table D.1:** The results of applying our verification algorithm to 18 UCI datasets.

| | | | | largest | all candidates satisfy | both 1st and 2nd | |
|---|---|---|---|---|---|---|---|
| Dataset | size | # features | # labels | candidate size | the 1st condition | conditions were verified | trade-off |
| Handwritten Digits | 1797 | 64 | 10 | 4 | ✓ | ✘ | YES |
| Haberman's Survival | 306 | 3 | 2 | 2 | ✓ | ✘ | YES |
| Letter Recognition | 20000 | 16 | 26 | 2 | ✓ | ✘ | YES |
| Somerville Happiness | 143 | 6 | 2 | 2 | ✓ | ✘ | YES |
| Vehicle Silhouettes | 846 | 18 | 4 | 2 | ✓ | ✘ | YES |
| Caesarian | 80 | 5 | 2 | 3 | ✓ | ✘ | YES |
| Musk (Version 1) | 476 | 166 | 2 | 1 | ✓ | ✘ | YES |
| Musk (Version 2) | 6598 | 166 | 2 | 1 | ✓ | ✘ | YES |
| Optical Digits | 3823 | 64 | 10 | 2 | ✓ | ✘ | YES |
| Pen-Based Digits | 7494 | 16 | 10 | 2 | ✓ | ✘ | YES |
| Mushroom | 5644 | 22 | 2 | 3 | ✓ | ✘ | YES |
| Nursery | 12960 | 8 | 5 | 8 | ✘ | ✓ | YES |
| Census Income | 32561 | 14 | 2 | 3 | ✓ | ✘ | YES |
| Chess | 28056 | 6 | 18 | 5 | ✓ | ✘ | YES |
| Contraceptive Method | 1473 | 9 | 3 | 4 | ✓ | ✘ | YES |
| Balance Scale | 625 | 4 | 3 | 3 | ✓ | ✘ | YES |
| Breast Cancer | 277 | 9 | 2 | 4 | ✓ | ✘ | YES |
| Car Evaluation | 1728 | 6 | 4 | 4 | ✓ | ✘ | YES |

# Appendix E

# A Non-optimal Classifier That Satisfies All the Three Properties

As we discussed in Section 4.7, one may give up optimality in order to achieve a classifier that simultaneously satisfies all the three properties defined in Section 4.4. In this section, we present an example of such a non-optimal classifier for the dataset in Table 4.1.

We use a probabilistic classifier for our discussion. Let $\mathbf{x}.f_i$ denote the value of feature $f_i$ in data point $\mathbf{x}$. Now consider the classifier defined by equation (E.1). This classifier first selects a linear classifier based on the set of known feature values $S$. Then it returns a binary label using an additional randomization step.

Table E.1 shows the accuracies of the predictions made by this classifier for each data point and each feature set. The values in the table are calculated using equation (4.1). First observe that by using feature set $\{f_1, f_2\}$ for all the data points, the classifier satisfies fair privacy. Furthermore, the accuracy of the classifier for all data points using this feature set is $\frac{4}{5}$, meaning that fair accuracy is also satisfied. Finally, we observe that using any subset of $\{f_1, f_2\}$ will result in a lower prediction accuracy for all the data points in the dataset, thus the need-to-know principle is also satisfied.

**Table E.1:** Accuracies of the predictions made by the classifier in equation (E.1).

| | $\emptyset$ | $\{f1\}$ | $\{f2\}$ | $\{f1, f2\}$ |
|---|---|---|---|---|
| $\mathbf{x}_1$ | 1/2 | 5/12 | 3/4 | 4/5 |
| $\mathbf{x}_2$ | 1/2 | 5/12 | 3/4 | 4/5 |
| $\mathbf{x}_3$ | 1/2 | 5/12 | 1/2 | 4/5 |
| $\mathbf{x}_4$ | 1/2 | 3/4 | 1/2 | 4/5 |

$$\hat{Y}(\Omega_S(\mathbf{x})) = \begin{cases} S = \{f_1, f_2\} & \begin{cases} \mathbf{x}.f_2 - \mathbf{x}.f_1 \geq 2 & + \quad w.p.\frac{4}{5} \\ otherwise & - \quad w.p.\frac{4}{5} \end{cases} \\ S = \{f_1\} & \begin{cases} \mathbf{x}.f_1 \geq 1 & - \quad w.p.\frac{3}{4} \\ otherwise & + \quad w.p.\frac{3}{4} \end{cases} \\ S = \{f_2\} & \begin{cases} \mathbf{x}.f_1 \geq 2 & + \quad w.p.\frac{3}{4} \\ otherwise & - \quad w.p.\frac{3}{4} \end{cases} \\ S = \emptyset & \begin{cases} + & w.p.\frac{1}{2} \\ - & w.p.\frac{1}{2} \end{cases} \end{cases} \tag{E.1}$$

In order to see that the classifier defined by equation (E.1) is not optimal, notice that our optimality definition requires the classifier to be optimal for all data points and using any subset $S \subset F$. However, using $\{f_1, f_2\}$ we see that the accuracy of the classifier for all the data points is $\frac{4}{5}$ while all the data points are distinguishable using $\{f_1, f_2\}$, i.e., the accuracy of an optimal classifier using $\{f_1, f_2\}$ is 1 for all data points in the dataset.

This example illustrates an interesting direction for future research, which can be stated as the following question: *“how much one needs to compromise on optimality in order to simultaneously achieve all the three socially desirable properties introduced in this paper?”*

# Appendix F

# Exploration Strategies

This section presents the Pseudocodes for the four exploration algorithms introduced in Section 5.6

**Algorithm 5:** Thompson Sampling (TS)

```
Input: Success and failure counters
for $f_j \in F$ do
    for $b$ in $\mathcal{X}_j$ do
        $\theta_j^b \sim Beta(x; S_j^b + a, F_j^b + c)$
$j^*, b^* = \operatorname{argmin}_{j,b} \theta_j^b$
return $(f_{j^*}, b^*)$
```

**Algorithm 6:** Top-Two Thompson Sampling (TTTS)

```
Input: Success and failure counters
for $f_j \in F$ do
    for $b$ in $\mathcal{X}_j$ do
        $\theta_j^b \sim Beta(x; S_j^b + a, F_j^b + c)$
$j^*, b^* = \text{argmin}_{j,b}\, \theta_j^b$
$K \sim Bernoulli(1/2)$
if $K=1$ then
    return $(f_{j^*}, b^*)$
else
    repeat
        for $f_j \in F$ do
            for $b$ in $\mathcal{X}_j$ do
                $\theta_j^b \sim Beta(x; S_j^b + a, F_j^b + c)$
        $\tilde{j}, \tilde{b} = \text{argmin}_{j,b}\, \theta_j^b$
    until $(\tilde{j}, \tilde{b}) \neq (j^*, b^*)$;
    return $(f_{\tilde{j}}, \tilde{b})$
```

**Algorithm 7:** Greedy

```
Input: Success and failure counters, $\beta^*$
for $f_j \in F$ do
    for $b$ in $\mathcal{X}_j$ do
        $p_j^b = F_{Beta}(\beta^*; S_j^b + a, F_j^b + c)$
$j^*, b^* = \text{argmax}_{j,b}\, p_j^b$
return $(f_{j^*}, b^*)$
```

**Algorithm 8:** Probability Matching (PM)

```
Input: Success and failure counters, $\beta^*$
for $f_j \in F$ do
    for $b$ in $\mathcal{X}_j$ do
        $p_j^b = F_{Beta}(\beta^*; S_j^b + a, F_j^b + c)$
Randomly choose an arm $(f_j, b)$ with
  probability $\frac{p_j^b}{\sum\limits_{(f_j,b)} p_j^b}$
return $(f_j, b)$
```

# References

Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., and Wallach, H. (2018). A reductions approach to fair classification. *arXiv preprint arXiv:1803.02453.*

Aggarwal, C. C. (2005). On k-anonymity and the curse of dimensionality. In *Proceedings of the 31st international conference on Very large data bases*, pages 901–909. VLDB Endowment.

Audibert, J.-Y., Bubeck, S., and Munos, R. (2010). Best arm identification in multi-armed bandits. In *COLT*, pages 41–53.

Bagdasaryan, E., Poursaeed, O., and Shmatikov, V. (2019). Differential privacy has disparate impact on model accuracy. In *Advances in Neural Information Processing Systems*, pages 15453–15462.

Barocas, S. and Selbst, A. D. (2016). Big data's disparate impact. *Cal. L. Rev.*, 104:671.

Berk, R., Heidari, H., Jabbari, S., Joseph, M., Kearns, M., Morgenstern, J., Neel, S., and Roth, A. (2017). A convex framework for fair regression. *arXiv preprint arXiv:1706.02409.*

Beutel, A., Chen, J., Zhao, Z., and Chi, E. H. (2017a). Data decisions and theoretical implications when adversarially learning fair representations. *arXiv preprint arXiv:1707.00075.*

Beutel, A., Chi, E. H., Cheng, Z., Pham, H., and Anderson, J. (2017b). Beyond globally optimal: Focused learning for improved recommendations. In *Proceedings of the 26th International Conference on World Wide Web*, pages 203–212.

Biega, A. J., Gummadi, K. P., and Weikum, G. (2018). Equity of attention: Amortizing individual fairness in rankings. *arXiv preprint arXiv:1805.01788.*

Biega, A. J., Potash, P., Daumé, III, H., Diaz, F., and Finck, M. (2020). Operationalizing the legal principle of data minimization for personalization. In *The 43rd International ACM SIGIR Conference on Research & Development in Information Retrieval.*

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. (2013). Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer.

Boyd, D. and Crawford, K. (2012). Critical questions for big data: Provocations for a cultural, technological, and scholarly phenomenon. *Information, communication & society*, 15(5):662–679.

Bubeck, S., Munos, R., and Stoltz, G. (2009). Pure exploration in multi-armed bandits problems. In *International conference on Algorithmic learning theory*, pages 23–37. Springer.

Buolamwini, J. and Gebru, T. (2018). Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pages 77–91. PMLR.

Burke, R., Sonboli, N., and Ordonez-Gauger, A. (2018). Balanced neighborhoods for multi-sided fairness in recommendation. In *Conference on Fairness, Accountability and Transparency*, pages 202–214.

Calmon, F., Wei, D., Vinzamuri, B., Ramamurthy, K. N., and Varshney, K. R. (2017). Optimized pre-processing for discrimination prevention. In *Advances in Neural Information Processing Systems*, pages 3992–4001.

Canetti, R., Cohen, A., Dikkala, N., Ramnarayan, G., Scheffler, S., and Smith, A. (2019). From soft classifiers to hard decisions: How fair can we be? In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 309–318. ACM.

Celis, L. E., Deshpande, A., Kathuria, T., and Vishnoi, N. K. (2016). How to be fair and diverse? *arXiv preprint arXiv:1610.07183*.

Chai, X., Deng, L., Yang, Q., and Ling, C. X. (2004). Test-cost sensitive naive bayes classification. In *Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 51–58. IEEE.

Chaney, A. J., Stewart, B. M., and Engelhardt, B. E. (2017). How algorithmic confounding in recommendation systems increases homogeneity and decreases utility. *arXiv preprint arXiv:1710.11214*.

Chapelle, O. and Li, L. (2011). An empirical evaluation of thompson sampling. *Advances in neural information processing systems*, 24:2249–2257.

Chen, H., Zhang, H., Si, S., Li, Y., Boning, D., and Hsieh, C.-J. (2019). Robustness verification of tree-based models. In *Advances in Neural Information Processing Systems*, pages 12317–12328.

Chouldechova, A. (2017). Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163.

Chouldechova, A. and Roth, A. (2020). A snapshot of the frontiers of fairness in machine learning. *Communications of the ACM*, 63(5):82–89.

Cohen, A. and Nissim, K. (2020). Towards formalizing the gdpr's notion of singling out. *Proceedings of the National Academy of Sciences*, 117(15):8344–8352.

Corbett-Davies, S., Pierson, E., Feller, A., Goel, S., and Huq, A. (2017). Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 797–806. ACM.

Cummings, R., Gupta, V., Kimpara, D., and Morgenstern, J. (2019). On the compatibility of privacy and fairness. In *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*, UMAP'19 Adjunct, pages 309–315, New York, NY, USA. ACM.

Dandekar, P., Goel, A., and Lee, D. T. (2013). Biased assimilation, homophily, and the dynamics of polarization. *Proceedings of the National Academy of Sciences*, 110(15):5791–5796.

Datta, A., Datta, A., Procaccia, A. D., and Zick, Y. (2015). Influence in classification via cooperative game theory. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Dua, D. and Graff, C. (2017a). UCI machine learning repository.

Dua, D. and Graff, C. (2017b). UCI machine learning repository.

Dwork, C., Hardt, M., Pitassi, T., Reingold, O., and Zemel, R. (2012). Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226. ACM.

Dwork, C., Roth, A., et al. (2014). The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407.

Dwyer, K. and Holte, R. (2007). Decision tree instability and active learning. In *European conference on machine learning*, pages 128–139. Springer.

Early, K., Fienberg, S. E., and Mankoff, J. (2016). Test time feature ordering with focus: Interactive predictions with minimal user burden. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 992–1003. ACM.

Ekstrand, M. D., Joshaghani, R., and Mehrpouyan, H. (2018). Privacy for all: ensuring fair and equitable privacy protections. In *Conference on Fairness, Accountability and Transparency*, pages 35–47.

Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., and Venkatasubramanian, S. (2015). Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268. ACM.

Finck, M. and Biega, A. (2021). Reviving purpose limitation and data minimisation in personalisation, profiling and decision-making systems. *Max Planck Institute for Innovation & Competition Research Paper*, (21-04).

Galdon Clavell, G., Martín Zamorano, M., Castillo, C., Smith, O., and Matic, A. (2020). Auditing algorithms: On lessons learned and the risks of data minimization. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 265–271.

Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., and Vechev, M. (2018). Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE.

Ginart, A., Guan, M., Valiant, G., and Zou, J. (2019). Making ai forget you: Data deletion in machine learning. *Advances in neural information processing systems*.

Goddard, M. (2017). The eu general data protection regulation (gdpr): European regulation that has a global impact. *International Journal of Market Research*, 59(6):703–705.

Grgic-Hlaca, N., Redmiles, E. M., Gummadi, K. P., and Weller, A. (2018). Human perceptions of fairness in algorithmic decision making: A case study of criminal risk prediction. In *Proceedings of the 2018 World Wide Web Conference*.

Hajian, S., Domingo-Ferrer, J., Monreale, A., Pedreschi, D., and Giannotti, F. (2015). Discrimination-and privacy-aware patterns. *Data Mining and Knowledge Discovery*, 29(6):1733–1782.

Hajian, S., Monreale, A., Pedreschi, D., Domingo-Ferrer, J., and Giannotti, F. (2012). Injecting discrimination and privacy awareness into pattern discovery. In *2012 IEEE 12th International Conference on Data Mining Workshops*, pages 360–369. IEEE.

Hannak, A., Sapiezynski, P., Molavi Kakhki, A., Krishnamurthy, B., Lazer, D., Mislove, A., and Wilson, C. (2013). Measuring personalization of web search. In

*Proceedings of the 22nd international conference on World Wide Web*, pages 527–538. ACM.

Hardt, M. (2014). Understanding alternating minimization for matrix completion. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 651–660. IEEE.

Hardt, M., Price, E., Srebro, N., et al. (2016). Equality of opportunity in supervised learning. In *Advances in neural information processing systems*, pages 3315–3323.

Harper, F. M. and Konstan, J. A. (2016). The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19.

Hastie, T., Mazumder, R., Lee, J. D., and Zadeh, R. (2015). Matrix completion and low-rank svd via fast alternating least squares. *The Journal of Machine Learning Research*, 16(1):3367–3402.

House, W. (2016). Big data: A report on algorithmic systems, opportunity, and civil rights. executive office of the president.

Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B. I., and Tygar, J. (2011). Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, pages 43–58. ACM.

Jagielski, M., Kearns, M., Mao, J., Oprea, A., Roth, A., Sharifi-Malvajerdi, S., and Ullman, J. (2019). Differentially private fair learning. In *International Conference on Machine Learning*, pages 3000–3008.

Jedor, M., Louedec, J., and Perchet, V. (2019). Categorized bandits. In *Advances in Neural Information Processing Systems*.

Kaminski, M. E. (2019). The right to explanation, explained. *Berkeley Tech. LJ*, 34:189.

Kamiran, F. and Calders, T. (2012). Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, 33(1):1–33.

Kamishima, T. and Akaho, S. (2017). Considerations on recommendation independence for a find-good-items task. In *FATREC Workshop on Responsible Recommendation Proceedings*.

Kamishima, T., Akaho, S., Asoh, H., and Sakuma, J. (2012). Enhancement of the neutrality in recommendation. In *Decisions@ RecSys*, pages 8–14.

Kamishima, T., Akaho, S., Asoh, H., and Sakuma, J. (2018). Recommendation independence. In *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81, pages 187–201, New York, NY, USA. PMLR.

Kamishima, T., Akaho, S., and Sakuma, J. (2011). Fairness-aware learning through regularization approach. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 643–650. IEEE.

Kantchelian, A., Tygar, J. D., and Joseph, A. (2016). Evasion and hardening of tree ensemble classifiers. In *International Conference on Machine Learning*, pages 2387–2396. PMLR.

Kleinberg, J., Mullainathan, S., and Raghavan, M. (2016). Inherent trade-offs in the fair determination of risk scores. *arXiv preprint arXiv:1609.05807*.

Koops, B.-J. (2011). Forgetting footprints, shunning shadows: A critical analysis of the right to be forgotten in big data practice. *SCRIPTed*, 8:229.

Kumar, S., Gao, H., Wang, C., Chang, K. C.-C., and Sundaram, H. (2019). Hierarchical multi-armed bandits for discovering hidden populations. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 145–153.

Larson, J., Mattu, S., Kirchner, L., and Angwin, J. (2016). How we analyzed the compas recidivism algorithm. *ProPublica (5 2016)*, 9.

Laskov, P. et al. (2014). Practical evasion of a learning-based classifier: A case study. In *2014 IEEE symposium on security and privacy*, pages 197–211. IEEE.

Lattimore, T. and Szepesvári, C. (2020). *Bandit algorithms*. Cambridge University Press.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Li, B., Wang, Y., Singh, A., and Vorobeychik, Y. (2016). Data poisoning attacks on factorization-based collaborative filtering. In *Advances in neural information processing systems*, pages 1885–1893.

Li, R.-H. and Belford, G. G. (2002). Instability of decision tree classification algorithms. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 570–575.

Ling, C. X., Yang, Q., Wang, J., and Zhang, S. (2004). Decision trees with minimal costs. In *Proceedings of the twenty-first international conference on Machine learning*, page 69. ACM.

Maliah, S. and Shani, G. (2018). Mdp-based cost sensitive classification using decision trees. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Matakos, A., Terzi, E., and Tsaparas, P. (2017). Measuring and moderating opinion polarization in social networks. *Data Mining and Knowledge Discovery*, 31(5):1480–1505.

Mei, S. and Zhu, X. (2015). Using machine teaching to identify optimal training-set attacks on machine learners.

Melis, L., Song, C., De Cristofaro, E., and Shmatikov, V. (2019). Exploiting unintended feature leakage in collaborative learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 691–706. IEEE.

Munoz, C., Director, D. P. C., of Science, M. U. C. T. O. S. O., Policy)), T., for Data Policy, D. D. C. T. O., of Science, C. D. S. P. O., and Policy)), T. (2016). *Big data: A report on algorithmic systems, opportunity, and civil rights*. Executive Office of the President.

Nelson, B., Rubinstein, B. I., Huang, L., Joseph, A. D., and Tygar, J. (2010). Classifier evasion: Models and open problems. In *International Workshop on Privacy and Security Issues in Data Mining and Machine Learning*, pages 92–98. Springer.

Nicas, J. (2018). How youtube drives people to the internet's darkest corners. *Wall Street Journal*.

Noriega-Campero, A., Bakker, M. A., Garcia-Bulle, B., and Pentland, A. (2019). Active fairness in algorithmic decision making. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 77–83. ACM.

O'Callaghan, D., Greene, D., Conway, M., Carthy, J., and Cunningham, P. (2015). Down the (white) rabbit hole: The extreme right and online recommender systems. *Social Science Computer Review*, 33(4):459–478.

of Health, U. D., Services, H., et al. (2003). Health information privacy. the privacy rule.

Pariser, E. (2011). *The Filter Bubble: What the Internet Is Hiding from You*. Penguin Group USA.

Pattuk, E., Kantarcioglu, M., Ulusoy, H., and Malin, B. (2015). Privacy-aware dynamic feature selection. In *2015 IEEE 31st international conference on data engineering*, pages 78–88. IEEE.

Politou, E., Alepis, E., and Patsakis, C. (2018). Forgetting personal data and revoking consent under the gdpr: Challenges and proposed solutions. *Journal of Cybersecurity*, 4(1):tyy001.

Pratesi, F., Gabrielli, L., Cintia, P., Monreale, A., and Giannotti, F. (2020). Primule: Privacy risk mitigation for user profiles. *Data & Knowledge Engineering*, 125:101786.

Pratesi, F., Monreale, A., Trasarti, R., Giannotti, F., Pedreschi, D., and Yanagihara, T. (2018). Prudence: a system for assessing privacy risk vs utility in data sharing ecosystems. *Transactions on Data Privacy*, 11(2):139–167.

Raghunathan, A., Steinhardt, J., and Liang, P. (2018). Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*.

Rastegarpanah, B., Crovella, M., and Gummadi, K. P. (2020). Fair inputs and fair outputs: The incompatibility of fairness in privacy and accuracy. In *Adjunct Publication of the 28th ACM Conference on User Modeling, Adaptation and Personalization*, pages 260–267.

Regulation, P. (2016). Regulation (eu) 2016/679 of the european parliament and of the council. *REGULATION (EU)*, 679:2016.

Reidenberg, J. R. (1994). Setting standards for fair information practice in the us private sector. *Iowa L. Rev.*, 80:497.

Romei, A. and Ruggieri, S. (2014). A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review*, 29(5):582–638.

Ruggieri, S., Hajian, S., Kamiran, F., and Zhang, X. (2014). Anti-discrimination analysis using privacy attack strategies. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 694–710. Springer.

Russo, D. (2016). Simple bayesian algorithms for best arm identification. In *Conference on Learning Theory*, pages 1417–1418.

Russo, D. and Van Roy, B. (2014). Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243.

Russo, D., Van Roy, B., Kazerouni, A., Osband, I., and Wen, Z. (2017). A tutorial on thompson sampling. *arXiv preprint arXiv:1707.02038*.

Shim, H., Hwang, S. J., and Yang, E. (2018). Joint active feature acquisition and classification with variable-size set encoding. In *Advances in Neural Information Processing Systems*, pages 1375–1385.

Singh, A. and Joachims, T. (2018). Fairness of exposure in rankings. *arXiv preprint arXiv:1802.07281*.

Singh, G., Gehr, T., Mirman, M., Püschel, M., and Vechev, M. (2018). Fast and effective robustness certification. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31, pages 10802–10813. Curran Associates, Inc.

Speicher, T., Heidari, H., Grgic-Hlaca, N., Gummadi, K. P., Singla, A., Weller, A., and Zafar, M. B. (2018). A unified approach to quantifying algorithmic unfairness: Measuring individual & group unfairness via inequality indices. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2239–2248. ACM.

Steinhardt, J., Koh, P. W., and Liang, P. (2017). Certified defenses for data poisoning attacks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 35203532.

Stinson, D. R. (2005). *Cryptography: theory and practice.* Chapman and Hall/CRC.

Sweeney, L. (2013). Discrimination in online ad delivery. *Queue*, 11(3):10–29.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Tavani, H. T. (2007). Philosophical theories of privacy: Implications for an adequate online privacy policy. *Metaphilosophy*, 38(1):1–22.

Trapeznikov, K. and Saligrama, V. (2013). Supervised sequential classification under budget constraints. In *Artificial Intelligence and Statistics*, pages 581–589.

Utz, C., Degeling, M., Fahl, S., Schaub, F., and Holz, T. (2019). (un) informed consent: Studying gdpr consent notices in the field. In *Proceedings of the 2019 acm sigsac conference on computer and communications security*, pages 973–990.

Verma, S. and Rubin, J. (2018). Fairness definitions explained. In *2018 IEEE/ACM International Workshop on Software Fairness (FairWare)*, pages 1–7. IEEE.

Voigt, P. and Von dem Bussche, A. (2017). The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing.*

Wang, B. and Gong, N. Z. (2018). Stealing hyperparameters in machine learning. *arXiv preprint arXiv:1802.05351.*

Wang, Y., Audibert, J.-Y., and Munos, R. (2008). Infinitely many-armed bandits. In *Advances in Neural Information Processing Systems.*

Weng, L., Zhang, H., Chen, H., Song, Z., Hsieh, C.-J., Daniel, L., Boning, D., and Dhillon, I. (2018). Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pages 5276–5285. PMLR.

Wong, E. and Kolter, Z. (2018). Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295. PMLR.

Wong, E., Schmidt, F., Metzen, J. H., and Kolter, J. Z. (2018). Scaling provable adversarial defenses. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31, pages 8400–8409. Curran Associates, Inc.

Yang, J. and Honavar, V. (1998). Feature subset selection using a genetic algorithm. In *Feature extraction, construction and selection*, pages 117–136. Springer.

Yao, S. and Huang, B. (2017). Beyond parity: Fairness objectives for collaborative filtering. In *Advances in Neural Information Processing Systems*, pages 2925–2934.

Yue, Y., Hong, S. A., and Guestrin, C. (2012). Hierarchical exploration for accelerating contextual bandits. *arXiv preprint arXiv:1206.6454*.

Zafar, B., Valera, I., Gomez-Rodriguez, M., and Gummadi, K. P. (2017a). Training fair classifiers. In *AISTATS'17: 20th International Conference on Artificial Intelligence and Statistics*.

Zafar, M. B., Valera, I., Gomez Rodriguez, M., and Gummadi, K. P. (2017b). Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1171–1180.

Zafar, M. B., Valera, I., Gomez Rodriguez, M., and Gummadi, K. P. (2017c). Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1171–1180.

Zafar, M. B., Valera, I., Rogriguez, M. G., and Gummadi, K. P. (2017d). Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pages 962–970.

Zehlike, M., Bonchi, F., Castillo, C., Hajian, S., Megahed, M., and Baeza-Yates, R. (2017). Fa* ir: A fair top-k ranking algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1569–1578. ACM.

Zemel, R., Wu, Y., Swersky, K., Pitassi, T., and Dwork, C. (2013). Learning fair representations. In *International Conference on Machine Learning*, pages 325–333.

Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. (2018). Efficient neural network robustness certification with general activation functions. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31, pages 4939–4948. Curran Associates, Inc.

# CURRICULUM VITAE

[redacted]