

UNIVERSITY OF THESSALY
SCHOOL OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Fairness in Machine Learning: Seldonian Approaches

Diploma Thesis

Eleni Georgantzi

Supervisor: Aspasia Daskalopulu

September 2025



UNIVERSITY OF THESSALY
SCHOOL OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Fairness in Machine Learning: Seldonian Approaches

Diploma Thesis

Eleni Georgantzi

Supervisor: Aspasia Daskalopulu

September 2025



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

**Δικαιοσύνη στη Μηχανική Μάθηση: Προσεγγίσεις τύπου
Seldon**

Διπλωματική Εργασία

Ελένη Γεωργαντζή

Επιβλέπων/πουσα: Ασπασία Δασκαλοπούλου

Σεπτέμβριος 2025

Approved by the Examination Committee:

Supervisor **Aspassia Daskalopulu**

Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Michael Vassilakopoulos**

Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Alexandros Chroneos**

Professor, Department of Electrical and Computer Engineering, University of Thessaly

Acknowledgements

This thesis was conducted as part of the Erasmus+ exchange program during my studies at the University of Thessaly, in the department of Electrical and Computer Engineering. I would like to express my sincere gratitude to Professor Alberto Castellini at the Department of Computer Science, University degli Studi di Verona, for his valuable supervision, constructive feedback, and continuous support throughout the duration of this project.

I would also like to thank Professor Aspasia Daskalopulu at the Department of Electrical and Computer Engineering, University of Thessaly, for her academic supervision and guidance during my undergraduate studies.

I am grateful to the academic and administrative staff of both institutions for providing a supportive and stimulating learning environment.

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Eleni Georgantzi

Diploma Thesis

Fairness in Machine Learning: Seldonian Approaches

Eleni Georgantzi

Abstract

This thesis investigates the development and evaluation of machine learning models that enforce fairness constraints with rigorous statistical guarantees, focusing on regression tasks. It centers on the application of the Seldonian framework [1], specifically the Non-Discriminating Linear Regression (NDLR) and Quasi-Non-Discriminating Linear Regression (QNDLR) algorithms, designed to reduce algorithmic bias in predictive modeling. Building upon the foundational literature of the Seldonian framework, an initial phase involved thoroughly understanding and reproducing key supplementary materials to ensure a solid grasp of the underlying principles. The core contribution of this work lies in the full implementation of NDLR and QNDLR algorithms in Python, followed by extensive testing on synthetic datasets to validate their effectiveness in maintaining fairness while achieving competitive predictive accuracy. A crucial advancement in this thesis is the application of QNDLR to real-life data, specifically examining its performance on the UCI Law School dataset. This real-world evaluation highlights the practical benefits and challenges of the Seldonian approach outside synthetic scenarios. Furthermore, the thesis includes a detailed comparison of QNDLR against the GridSearchReduction method, demonstrating that QNDLR not only matches state-of-the-art performance in fairness and accuracy but also provides formal statistical guarantees, a key differentiator for trustworthy machine learning in sensitive applications

Keywords:

Machine Learning, Fairness, Regression, Seldonian Algorithms, Constrained Optimization, Statistical Guarantees

Διπλωματική Εργασία

Δικαιοσύνη στη Μηχανική Μάθηση: Προσεγγίσεις τύπου Seldon

Ελένη Γεωργαντζή

Περίληψη

Η παρούσα διπλωματική εργασία διερευνά την ανάπτυξη και αξιολόγηση μοντέλων μηχανικής μάθησης που επιβάλλουν περιορισμούς δικαιοσύνης με αυστηρές στατιστικές εγγυήσεις, εστιάζοντας σε εργασίες παλινδρόμησης. Επικεντρώνεται στην εφαρμογή του πλαισίου Seldonian [1], και συγκεκριμένα στους αλγόριθμους Non-Discriminating Linear Regression (NDLR) και Quasi-Non-Discriminating Linear Regression (QNDLR), σχεδιασμένους να μειώνουν την αλγοριθμική μεροληψία σε προγνωστικά μοντέλα.

Βασιζόμενη στη θεμελιώδη βιβλιογραφία του πλαισίου Seldonian, η αρχική φάση περιελάμβανε την εις βάθος κατανόηση και αναπαραγωγή βασικού συμπληρωματικού υλικού, προκειμένου να διασφαλιστεί η σωστή κατανόηση των βασικών αρχών. Η κύρια συνεισφορά της εργασίας αυτής είναι η πλήρης υλοποίηση των αλγορίθμων NDLR και QNDLR σε Python, ακολουθούμενη από εκτενείς δοκιμές σε συνθετικά σύνολα δεδομένων, για την επαλήθευση της αποτελεσματικότητάς τους στη διατήρηση της δικαιοσύνης, παράλληλα με την επίτευξη ανταγωνιστικής προβλεπτικής ακρίβειας.

Σημαντική πρόοδος της διπλωματικής αποτελεί η εφαρμογή του QNDLR σε πραγματικά δεδομένα, και συγκεκριμένα η αξιολόγησή του στο σύνολο δεδομένων UCI Law School. Αυτή η αξιολόγηση σε ρεαλιστικό περιβάλλον αναδεικνύει τα πρακτικά οφέλη και τις προκλήσεις της προσέγγισης Seldonian εκτός των συνθετικών σεναρίων. Επιπλέον, η εργασία περιλαμβάνει λεπτομερή σύγκριση του QNDLR με τη μέθοδο GridSearchReduction, αποδεικνύοντας ότι το QNDLR όχι μόνο ισοφαρίζει τις επιδόσεις state of art μοντέλων όσον αφορά τη δικαιοσύνη και την ακρίβεια, αλλά προσφέρει και επίσημες στατιστικές εγγυήσεις — παράγοντα-κλειδί για την ανάπτυξη αξιόπιστης μηχανικής μάθησης σε ευαίσθητες εφαρμογές.

Λέξεις-κλειδιά:

Μηχανική Μάθηση, Δικαιοσύνη, Παλινδρόμηση, Αλγόριθμοι Seldonian, Βελτιστοποίηση με Περιορισμούς, Στατιστικές Εγγυήσεις

Table of contents

Acknowledgements	ix
Abstract	xii
Περίληψη	xiii
Table of contents	xv
List of figures	xix
List of tables	xxi
1 Introduction	1
1.1 Thesis Subject	2
1.1.1 Contribution	3
1.2 Thesis Structure	3
2 Background and Related Work	5
2.1 Background	5
2.1.1 Fairness in Machine Learning and Discrimination Statistics	5
2.1.2 Hoeffding's Inequality	7
2.1.3 Student's t-Test	7
2.1.4 Bounded Group Loss as a Fairness Metric	8
2.1.5 AI Fairness 360 (AIF360)	9
2.1.6 Fairlearn	9
2.1.7 Grid Search Reduction for Fair Regression	10
2.2 Related Work	11

3	Methodology – Reproducing and Analyzing Regression Fairness	13
3.1	The illustrative Example	13
3.1.1	Understanding the Experimental Setup	14
3.1.2	Synthetic Data Generation	14
3.1.3	Least Squares Linear Regression	15
3.1.4	Discriminatory Behavior and the Discrimination Statistic	16
3.1.5	Implementation of Model Fitting and Overall Fairness Evaluation .	17
3.1.6	Visualization of Discrimination in Linear Regression Predictions .	18
3.2	Root causes and Potential Remedies for Undesirable Behavior	19
3.2.1	Understanding Root Causes of Undesirable Behavior	19
3.2.2	Potential Remedies	20
3.2.3	Experimental Results: Soft Constraints and Discrimination Mitigation	21
3.2.4	Recreation of Soft constraint experiment	23
4	Methodology - The Seldonian Framework	31
4.1	Towards Fairer Models: The Seldonian Approach	31
4.2	The Seldonian Framework	32
4.2.1	Motivation and Limitations of Prior Approaches	32
4.2.2	Seldonian Optimization Problem (SOP)	32
4.2.3	Interpretation and Practical Impact	33
4.2.4	Conceptual Shift in Algorithm Design	34
4.3	Seldonian and Quasi Seldonian Algorithms	34
4.3.1	NDLR: Non Discriminatory Linear Regression	36
4.3.2	QNDLR(λ): Quasi Non Discriminatory Linear Regression	40
5	Results	47
5.1	Applying NDLR and QNDLR to the Illustrative Example	47
5.1.1	Experimental Setup	47
5.1.2	Results	48
5.1.3	NDLR & QNDLR Experiments	50
5.2	Seldonian Algorithms and Real Life Data	54

5.2.1	Experimental Setup	54
5.2.2	Results	56
5.2.3	Analysis of Results	57
5.3	QNDLR vs. Grid Search Reduction	58
5.3.1	Experimental Setup	58
5.3.2	QNDLR vs. GridSearchReduction - Results	61
6	Conclusion	63
6.1	Summary and Conclusions	63
6.2	Future Extensions	65
	Bibliography	67
	APPENDICES	71
A	Source Code Documentation	73
A.1	Code Navigation	73
A.2	Code-to-Thesis Mapping	75
A.3	Code Access	75
A.3.1	Dependencies:	75

List of figures

3.1	Figure S2 as presented in the paper	18
3.2	Recreation of Figure S2	19
3.3	Figure S4 as presented in [1]	22
3.4	Discrimination vs. λ	24
3.5	MSE vs. λ	24
3.6	Gradient computation was handled using TensorFlow's GradientTape	28
4.1	HoeffdingUpperBound(Z, b, δ) [1]	37
4.2	PredictHoeffding(Z, b, δ, k) [1]	37
4.3	HoeffdingCandidateObjective($\theta, D, \delta, \epsilon, b, k$) [1]	38
4.4	HoeffdingDiscrimUpperBound($\theta, D, \delta, \epsilon, b$) [1]	39
4.5	Non-Discriminatory Linear Regression (NDLR) [1]	39
4.6	TTestUpperBound(Z, δ) [1]	42
4.7	PredictTTest(Z, b, δ, k) [1]	42
4.8	TTestCandidateObjective($\theta, D, \delta, \epsilon, k, \lambda$) [1]	43
4.9	TTestDiscrimUpperBound($\theta, D, \delta, \epsilon$) [1]	44
4.10	Quasi-Non-Discriminatory Linear Regression (QNDLR if $\lambda = 0$ and QNDLR(λ) if $\lambda > 0$) [1]	44
5.1	5,000 randomly sampled points from D with the optimal regression line re- turned by NDLR.	48
5.2	Comparison of NDLR (green dotted) and least squares (black solid) regres- sion lines. Points are colored by sensitive group T	49
5.3	5,000 randomly sampled points from D with the optimal regression line re- turned by QNDLR.	50

5.4	Comparison of regression lines from NDLR (green dashed), QNDLR (blue dashed), and least squares (black solid). Points are colored by sensitive group T	50
5.5	Probability that NDLR and QNDLR return a solution as a function of training set size m , averaged over 100 trials per m	51
5.6	Probability that discrimination statistic exceeds $\epsilon = 0.1$ for Linear Regression, NDLR, and QNDLR across training sizes.	52
5.7	Mean discrimination statistic $d(\theta)$ for Linear Regression, NDLR, and QNDLR across training sizes.	53
5.8	Mean Squared Error for Linear Regression, NDLR, and QNDLR across training sizes.	54

List of tables

5.1	Performance and fairness comparison across methods.	62
A.1	Mapping of thesis experiments and code files	75

Chapter 1

Introduction

In recent years, there has been an increasing interest in using machine learning models for decision-making operations across different areas of finance, medicine, and education. These models, while robust for prediction, have a tendency to inherit or even amplify biases within the data. Therefore, fairness concerns and ethical use of autonomous decision systems have taken center stage. Ensuring predictive models are free from discrimination against protected groups has become a challenge for both researchers and practitioners.

A central issue here is the inherent trade-off between maximizing prediction accuracy as much as possible while maintaining fairness constraints. Classical regression techniques tend to favor minimizing prediction error at the cost of imbalances that may arise in various demographic groups. Fairness in classification problems is well-studied, whereas fairness in regression problems has remained relatively underexplored. There is a lack of comprehensive methods that address fairness while also giving statistical guarantees in regression tasks.

One of the central questions in algorithmic fairness is whether sensitive attributes such as race, gender, or age ought to be included in the training procedure. While it might seem intuitively reasonable to exclude these features so as to not explicitly discriminate—a notion often referred to as fairness through unawareness—it is often unsatisfactory. Discrimination can still remain because of correlations between sensitive and non-sensitive features, meaning models can be unfair even if they do not explicitly use the sensitive attribute [2, 3].

However, employing sensitive attributes during training makes models more capable of assessing and mitigating unfairness. This is important especially when computing group-level fairness metrics or enforcing fairness constraints, as it enables the model to detect and correct disparities between protected and unprotected groups [4] In most legal and ethical contexts,

though, direct application of sensitive features to the prediction process may be undesirable or even forbidden [5].

This thesis focuses on the Seldonian framework [1] as a novel approach to fairness in regression, in particular through the use of algorithms such as NDLR and QNDLR(λ), which also provide statistical guarantees on discrimination. The framework enforces fairness constraints based on group-level statistics, without directly including the sensitive attribute in the model's input features. This formulation helps building models that are both fair and compliant with practical constraints on sensitive data usage.

1.1 Thesis Subject

As previously discussed, the field of fairness in regression tasks remains significantly underdeveloped compared to its classification counterpart. This thesis seeks to contribute toward closing this gap by exploring, reproducing, and extending a principled framework for fair regression that offers probabilistic fairness guarantees.

To address the underdeveloped state of fairness in regression tasks, this thesis approaches machine learning unfairness as a form of undesirable behavior. It refines its focus through the lens of the paper [1], which frames fairness violations—such as group-based bias—as undesirable algorithmic outcomes. Special attention is given to the supplementary material of the paper, which presents a compelling case: it illustrates how standard models like linear regression can harbor hidden biases, critiques the limitations of conventional bias mitigation techniques, and introduces the Seldonian framework as a novel solution to these challenges.

Building on this foundation, the thesis reconstructs and analyzes the examples provided in the original work to verify and better understand its claims. It then presents the Seldonian framework in detail, elaborates on the conceptual underpinnings of Seldonian algorithms, and implements them in Python. These implementations are evaluated on both synthetic and real datasets, with performance and fairness outcomes benchmarked against state-of-the-art alternatives. Through this process, the thesis not only reproduces key results but also expands the empirical understanding of fairness-aware regression, contributing to the broader conversation on reliable and equitable machine learning systems.

1.1.1 Contribution

The contribution of this thesis is summarized as follows:

1. The foundational paper “*Preventing Undesirable Behavior of Intelligent Machines*” [1] and its supplementary material were thoroughly studied, with particular focus on bias characterization, the shortcomings of common mitigation methods, and the introduction of the Seldonian framework.
2. Key examples from the original work were faithfully recreated to validate and deepen understanding of fairness-related phenomena in regression models.
3. Core Seldonian algorithms (NDLR and QNDLR(λ)) were implemented in Python following the provided pseudocode, enabling practical experimentation.
4. The implemented Seldonian algorithms were rigorously tested for both fairness and predictive accuracy on synthetic and real-world datasets.
5. The QNDLR algorithm was compared against state-of-the-art regression and fairness methods, highlighting its advantages and limitations in various scenarios.

1.2 Thesis Structure

Background and related work on fairness in machine learning, with an emphasis on regression tasks, are discussed in Chapter 2. Chapter 3 focuses on reproducing and interpreting the supplementary material from the foundational paper [1], including recreating key examples, analyzing sources of bias, and reimplementing soft-constraint mitigation approaches. In Chapter 4, the thesis presents the development and evaluation of Seldonian algorithms. This includes introducing the theoretical framework, detailing the NDLR and QNDLR algorithms with their Python implementations, and describing the experimental setups used for assessment. Chapter 5 reports on the experimental results obtained from synthetic and real datasets, comparing the performance of the Seldonian algorithms against state-of-the-art fairness-aware regression methods. Finally, Chapter 6 concludes the thesis by summarizing the main contributions, discussing limitations, and suggesting avenues for future research.

Chapter 2

Background and Related Work

This chapter provides the background and related work relevant to the thesis. It introduces the key concepts, methods and prior research necessary to understand the approaches presented later.

2.1 Background

This section reviews the foundational concepts and tools essential for understanding and addressing fairness in machine learning. It begins with fairness definitions and discrimination statistics, followed by key statistical inequalities and tests that support fairness guarantees. Then, fairness metrics tailored for regression tasks are presented, along with prominent fairness toolkits and algorithms such as AI Fairness 360, Fairlearn, and Grid Search Reduction. This background provides the groundwork for the methodology and experiments detailed in later chapters.

2.1.1 Fairness in Machine Learning and Discrimination Statistics

Fairness in machine learning addresses concerns that predictive models may inadvertently replicate or even amplify societal biases present in training data. Discriminatory behavior in ML refers to systematic differences in model performance or predicted outcomes across different subpopulations, especially when distinguished by sensitive attributes such as gender, race, or age. These disparities can occur in the form of biased error rates, unequal predictions, or disparate treatment and impact. [5]

From a regulatory standpoint, the importance of fairness has been codified in laws such

as the General Data Protection Regulation (GDPR) in the European Union. Article 22 of the GDPR explicitly gives individuals the right not to be subject to decisions based solely on automated processing—including profiling—that significantly affect them. This includes requiring that automated systems be transparent, explainable, and non-discriminatory. As a result, algorithmic fairness is not only a technical objective but also a legal necessity in many domains such as finance, healthcare, and employment.

To address fairness issues, machine learning researchers have introduced formal fairness definitions and metrics, and developed algorithmic interventions that constrain or correct biased model behavior.

One central concept in operationalizing fairness is the use of *discrimination statistics*. These are mathematical functions or metrics that quantify the extent of unfairness in a model's behavior with respect to sensitive features. By computing discrimination statistics, we can measure how differently a model behaves for different groups, and use this information to guide model improvement or apply fairness constraints. [6].

There are various types of discrimination statistics, each aligned with different fairness definitions. For example, the *demographic disparity* (also known as statistical parity difference) measures the difference in favorable outcome rates between groups. Other metrics, such as *equalized odds* or *disparate impact*, focus on error rate disparities or ratios of treatment outcomes. In regression tasks, group-conditional residuals or group-specific prediction averages are often used to assess bias. [7]

While many of these metrics and fairness algorithms have been extensively developed for classification tasks, fairness in regression remains a relatively underexplored area. The continuous nature of regression targets introduces challenges in defining what constitutes a "fair" prediction and in applying existing fairness constraints. As a result, the fairness community currently lacks the same breadth of ready-to-use tools and standardized approaches for regression problems, highlighting the importance of advancing this area of research.

In practice, these statistics serve both diagnostic and prescriptive roles: they help identify whether a model exhibits discriminatory patterns, and they are used within algorithmic procedures to enforce fairness constraints. Ultimately, by integrating these metrics into training pipelines, we can work toward machine learning models that are not only accurate but also socially and ethically aligned.

2.1.2 Hoeffding's Inequality

In probability theory Hoeffding's inequality [8] provides a nonasymptotic, distribution-free upper bound on the probability that the empirical mean of independent, bounded random variables deviates from their expected value. Formally, let $X_1, X_2, \dots, X_n$ be independent random variables such that $X_i \in [a_i, b_i]$ almost surely. Then, for any $\epsilon > 0$,

$$\Pr \left(\left| \frac{1}{n} \sum_{i=1}^n X_i - \mathbb{E} \left[\frac{1}{n} \sum_{i=1}^n X_i \right] \right| \geq \epsilon \right) \leq 2 \exp \left(-\frac{2n^2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2} \right).$$

In the special case where all variables are uniformly bounded within the same interval $[a, b]$, the inequality simplifies to:

$$\Pr \left(\left| \frac{1}{n} \sum_{i=1}^n X_i - \mathbb{E} \left[\frac{1}{n} \sum_{i=1}^n X_i \right] \right| \geq \epsilon \right) \leq 2 \exp \left(-\frac{2n\epsilon^2}{(b-a)^2} \right).$$

More practically, hoeffding's inequality is utilized for approximating how near is the sample mean, to the actual mean, despite a finite number of samples available. This renders hoeffding's inequality highly useful in data driven situations where good decisions must be arrived at based on limited empirical evidence.

Hoeffding's inequality is highly generalizable in broad applicability in many areas such as statistical learning theory, decision making and algorithmic performance analysis because it imposes relatively weak assumptions, namely independence and boundedness only. Its main advantage is that it provides clear, finite-sample guarantees, enabling robust reasoning about uncertainty and generalization in empirical studies.

2.1.3 Student's t-Test

Student's t-Test is a classic hypothesis test for investigating if there's a noticeable difference between the two groups' means. [9] It is very useful especially when the population variance is not known and the sample size is small. The test uses t-distribution to account for the extra uncertainty that arises from the fact that the population variance has been estimated from the sample.

Let $X_1, X_2, \dots, X_n$ be an independently sampled from a normal population with unknown mean μ and unknown variance σ^2 . The sample mean $\bar{X}$ and sample standard deviation S are defined as

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i, \quad S = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2}.$$

The test statistic for testing the null hypothesis $H_0 : \mu = \mu_0$ is

$$t = \frac{\bar{X} - \mu_0}{S/\sqrt{n}},$$

which follows a Student's t -distribution with $n - 1$ degrees of freedom under the null hypothesis. This allows the computation of a p -value or critical values to assess the statistical significance of the observed difference.

Practically, the Student's t -test allows scientists to draw inferences regarding population means from limited data while keeping in mind the variability of the sample. It is used across all experiment and social sciences and other disciplines with small sample sizes and unknown population parameters. Essentially, it estimates how probable observed differences are by chance. The t -test plays a pivotal part in hypothesis testing and data-driven decision-making. Although it assumes normality of the population, the test is still robust even when the assumption has been broken to a certain degree, particularly if the sample size is large.

The student's t -test is essentially a basic statistical inference technique in general. It blends practical applicability with correct probabilistic reasoning and is at times employed as a baseline when pitted against more advanced statistical methods. It is used specifically within this paper for the application of Hoeffding's inequality in the QNDLR algorithm

2.1.4 Bounded Group Loss as a Fairness Metric

Bounded Group Loss (BGL) [10] is a fairness measure aimed at capturing disparities in predictive accuracy between protected groups. The group-specific loss functions are at center stage. The original context in which it was presented is fairness-aware learning, with the aim of being sure that no subgroup experiences disproportionately high predictive errors. That's a frequent concern in a lot of real-world applications where fairness is not negotiable.

More formally, BGL limits the maximum expected loss within any protected group to fall below a desired threshold, thus imposing a group-level parity on model performance.

$$\mathbb{E}[\ell(Y, f(X)) \mid A = a] \leq \zeta \quad \forall a [7]$$

This is highly important in regression settings, where large differences between prediction errors across two groups may indicate biased or unequal predictions.

BGL may be implemented with various loss functions such as the absolute loss or squared loss function. For instance, when employing the MAE (Mean Absolute Error), BGL tries to

limit the MAE difference between groups. This ensures that the model is neither overpredicting nor underpredicting for any single group consistently. A low MAE difference indicates balanced performance.

Implementing BGL constraints into learning algorithms leads to models with a balanced trade-off between fairness and prediction quality.

2.1.5 AI Fairness 360 (AIF360)

AIF360 is a broad open-source library developed by IBM Research that facilitates the identification, understanding and reduction of bias in Machine Learning models [11]. Released in 2018, it provides users with a unified interface to a variety of fairness metrics and mitigation algorithms applicable at different points of the ML pipeline, including dataset pre-processing, model training, and prediction post-processing.

It was deployed in 2018 and provides users with varied fairness metrics and mitigation algorithms to apply at various points in ML pipeline.

It provides not only data pre-processing, model training, and prediction post-processing functionality but also reproducible experiments and standardized implementations of fairness algorithms.

The contribution of AIF360 is crucial given the fact that it is among the first and most complete toolkits dedicated to fairness in ML.

It works towards bridging the gap between theoretical fairness and practice, enabling researchers to seamlessly include fairness considerations into their pipelines [12]. The toolkit has plenty of citations and is used in projects and scientific literature, facts that speak to its status as a respected and authoritative standard for fairness research

2.1.6 Fairlearn

Fairlearn is an open-source Python library developed by Microsoft Research with the objective to measure and improve machine learning model fairness [13]. Launched in 2018, Fairlearn provides a framework that integrates fairness measurement and mitigation algorithms. The framework focuses mainly on in-processing and post-processing approaches.

Various fairness metrics and reduction-based algorithms, such as Grid Search Reduction [14] are offered to users. Fairlearn focuses on measuring bias and ensuring fairness through constrained optimization, making it an advanced approach in fair machine learning.

The appeal of Fairlearn is that it finds a balance between theoretical accuracy and usability. It integrates well with scikit-learn and comes with high documentation and examples which allow researchers to integrate fairness measures and protections against bias throughout the model development process. It has a wide applicability in academic studies and real-world applications fact that highlights its relevance as a first-order tool to advance fair and ethical AI.

2.1.7 Grid Search Reduction for Fair Regression

The Grid Search Reduction algorithm is a robust in-processing technique for enacting fairness constraints on regression and classification tasks. Originally proposed by Agarwal et al. (2018) [14], algorithm converts a fairness-constrained learning task into a sequence of cost-sensitive learning tasks. Grid search is used across a set of candidate models to identify the model having the smallest prediction error and satisfying fairness requirements, e.g., BGL.

The Grid Search Reduction algorithm has been implemented and codified in the Fairlearn library. While, it has also been integrated in the AIF360 package to make it compatible with scikit-learn data sets and APIs. Such integration enables the implementation of Grid Search algorithm along with other fairness algorithms and data sets offered by AIF360.

This algorithm is outstanding in its theoretical guarantees of fairness as well as in offering support for various fairness definitions and loss functions.

It is, unquestionably, an effective baseline to compare to when developing new fairness aware algorithms.

Algorithm 1 Grid Search Reduction [14, 15]

```

0: Define the fairness constraint (e.g.BGL)
0: Construct a grid of Lagrange multipliers  $\lambda_1, \lambda_2, \dots, \lambda_k$ 
0: Initialize an empty list to store models and their metrics
0: for each  $\lambda_i$  in the grid do
0:   Solve the weighted classification problem:
       Compute reweighting using reduction based on  $\lambda_i$  and the constraint
0:   Fit a new copy of the base estimator on the reweighted dataset
0:   Evaluate its prediction error and constraint violation
0:   Store the trained model and its metrics
0: end for
0: for each trained model do
0:   Compute selection objective:
       score =  $(1 - \text{constraint\_weight}) \cdot \text{error} + \text{constraint\_weight} \cdot \text{violation}$ 
0: end for
0: Select the model with the minimum score
0: Return the selected model for future predictions
=0

```

2.2 Related Work

Fairness in machine learning has become an increasingly important research area, aiming to ensure that algorithmic decisions do not produce or amplify social inequalities.

A foundational classification-oriented framework is the *reductions approach* proposed by Agarwal et al. in *A Reductions Approach to Fair Classification* [14], where fairness constraints are enforced by transforming the problem into a series of cost-sensitive learning tasks. This work laid the groundwork for scalable and flexible fairness-aware algorithms.

Building on this idea, the same authors extended the reductions methodology to regression settings in *Fair Regression: Quantitative Definitions and Reduction-based Algorithms* [7], introducing fairness definitions such as mean prediction parity and residual parity. They propose an exponentiated gradient method to optimize for both accuracy and fairness in regression, making it one of the few algorithmic contributions focused specifically on this task.

Another contribution is the work *Error Parity Fairness: Testing for Group Fairness in Regression Tasks* [16], which emphasizes statistical testing for fairness using residual-based group comparisons. It offers hypothesis-testing tools to detect disparities in prediction errors, highlighting challenges that arise uniquely in continuous outcome settings.

A broader survey is provided in *A Review on Fairness in Machine Learning* by Pessach and Shmueli [17], which classifies fairness definitions, measurement strategies, and algorithmic approaches. The review underscores that fairness in regression is relatively underdeveloped, both in terms of theoretical foundations and available tooling.

Several open-source toolkits have been developed to support fairness in machine learning by providing metrics and mitigation algorithms. Among them, **IBM's AI Fairness 360 (AIF360)** has emerged as a foundational toolkit widely used in both research and industry. Bellamy et al. [11] introduced AIF360 as an extensible framework covering multiple stages of the ML pipeline—pre-processing, in-processing, and post-processing—to address unwanted bias.

The fairness notions and mitigation methods implemented in AIF360 are grounded in seminal works on fairness definitions, such as Dwork et al.'s fairness through awareness [3] and Hardt et al.'s equality of opportunity framework [18]. Recent surveys and comparative studies recognize AIF360 as a standard baseline for fairness evaluation, citing its broad adoption and influence [19, 20].

Chapter 3

Methodology – Reproducing and Analyzing Regression Fairness

This chapter focuses on reproducing and interpreting the key components of the supplementary material accompanying the foundational paper “Preventing Undesirable Behavior of Intelligent Machines” by Thomas et al [1]. Its objective is to develop a deeper understanding of how fairness concerns emerge in regression tasks and to validate the original claims through hands-on reimplementation. The chapter begins by recreating the illustrative example from the paper, which reveals how standard linear regression models can exhibit systematic discrimination. It then investigates the root causes of this bias and examines why conventional mitigation strategies often fail to address it effectively. To further underscore these limitations, the chapter reimplements and analyzes a soft-constraint approach proposed in the original work. Together, these efforts establish both the motivation and the technical foundation for the introduction of Seldonian algorithms in the following chapter.

3.1 The illustrative Example

This section outlines the process followed to reproduce the illustrative example from the supplementary material of “Preventing Undesirable Behaviour of Intelligent Machines”. The example concerns the fairness of one of the most widely used and well-studied data analysis tools: linear regression. The goal of the example is to demonstrate how standard machine learning techniques can result in systematic discriminatory behaviour, even in synthetic and controlled environments. The following sections describe how this example was recreated

step by step, from generating synthetic data to implementing and evaluating the model and fairness metric.

3.1.1 Understanding the Experimental Setup

In this specific illustrative example, the goal is to predict the aptitude of job applicants based on a numerical representation of the quality of their resumes. Each applicant belongs to one of two demographic groups, referred to as Group A and Group B. For example, Group A could represent female applicants, and Group B male applicants. We refer to the group an applicant belongs to as their type. The training data includes three key components for each applicant: a single feature representing resume quality, a label representing actual aptitude, and a binary type indicator (0 for Group A, 1 for Group B), so each data point in the dataset consists of a triplet (x, y, t) . While the group information is available during training, it is assumed to be unknown at prediction time — reflecting a realistic constraint, since resumes typically do not contain explicit demographic information.

3.1.2 Synthetic Data Generation

To replicate the example, I produced a synthetic dataset that mirrors the statistical properties described in the original experiment

The data is constructed based on the following assumptions:

- **Demographic Balance:** Applicants come from two groups — A and B — with equal probability. The training dataset contains an equal number of applicants from each group.
- **Aptitude Distributions:** The true aptitude Y of each applicant is drawn from a normal distribution, with the mean depending on the group:
 - For group A (type = 0): $Y \sim \mathcal{N}(1, 1)$
 - For group B (type = 1): $Y \sim \mathcal{N}(-1, 1)$

This introduces a real difference in aptitude across groups — which is important, because the goal is not to enforce equal predictions, but rather to avoid systematic over- or underestimation of each group’s true ability.

- **Resume Quality (Feature X):** Resume quality X is generated as a noisy version of aptitude:

$$X \sim \mathcal{N}(Y, 1)$$

That is, each applicant's resume reflects their aptitude, but with additional noise drawn from a standard normal distribution. This models the idea that resumes are imperfect indicators of actual ability.

- **Sample Size:** Each training set contains $m = 1000$ applicants. For robustness, 10,000 independent training sets are generated using the above procedure.

The synthetic datasets were generated using a custom Python script leveraging the NumPy and Pandas libraries. NumPy was used to generate random samples from normal distributions to simulate the aptitude scores and resume quality features according to the defined statistical properties. Pandas was employed to organize this data into tabular form, facilitating easy manipulation and saving of the datasets.

To create each dataset, the script assigned group labels evenly between the two applicant types and generated the corresponding aptitude and resume quality values with the specified noise characteristics. This process was repeated 10,000 times to produce multiple independent datasets, which were then combined into a single dataset to be used for subsequent modeling and analysis.

3.1.3 Least Squares Linear Regression

As previously mentioned, the example focuses on evaluating fairness in linear regression. So, author's next step was to fit a linear estimator to each synthetic dataset using ordinary least squares. The model used is an affine function of the form:

$$\hat{y}(x) = \theta_1 x + \theta_2$$

where $\theta = [\theta_1, \theta_2] \in \mathbb{R}^2$ is the parameter vector learned by minimizing the mean squared error (MSE) between the predictions $\hat{y}(x)$ and the true aptitude values y .

Although the group membership T is available in the training data, it is excluded from the prediction function to reflect realistic deployment settings, where sensitive attributes such as gender or ethnicity are typically not available at inference time.

In my implementation, I applied ordinary least squares regression independently to each of the 10,000 synthetic datasets by fitting the function $\hat{y}(x) = \theta_1 x + \theta_2$. This fitting was performed using the `LinearRegression` class from the Python `scikit-learn` library. This produced 10,000 linear models, each corresponding to a different training set and ready to be evaluated for fairness using the discrimination statistic described in the following subsection. A more detailed examination of this application follows in a following section.

3.1.4 Discriminatory Behavior and the Discrimination Statistic

In the example, the discrimination statistic used is a measure of the difference in the average prediction errors across the two groups. More formally, it is defined as the absolute difference in the expected prediction error (residuals) between the groups:

$$d(\theta) := \mathbb{E}[u(y(X, \theta) - Y) \mid T = 0] - \mathbb{E}[u(y(X, \theta) - Y) \mid T = 1]$$

Here, $y(X, \theta)$ denotes the model's prediction for input X with parameters θ , and Y is the true value. $u : \mathbb{R} \rightarrow \mathbb{R}$ is a utility function and for simplicity reasons it is assumed to be the identity function. A large absolute value of $d(\theta)$ indicates biased predictions, whereas values near zero suggest that prediction errors are balanced fairly across groups. Large negative values of $d(\theta)$ indicate that the model tends to under-predict for group A (type 0) compared to group B (type 1), suggesting discrimination against type 0. Conversely, large positive values of $d(\theta)$ indicate over-prediction for group A relative to group B, implying discrimination against type 1.

To calculate the discrimination statistic $d(\theta)$ for each synthetic dataset, I defined a function named `compute_discrimination_statistic` that accepts two inputs: a dataset and a model. The function operates by first separating the dataset into two groups according to the sensitive attribute T . For each group, prediction errors are computed as the difference between the model's predicted values $y(X, \theta)$ and the true aptitude values Y . The mean prediction error within each group is then calculated, and the discrimination statistic $d(\theta)$ is obtained as the difference between these group-specific mean errors. This metric captures the degree to which the model's predictions systematically favor or disadvantage one group relative to the other, thereby quantifying potential bias. This computation was performed independently for each of the 10,000 synthetic datasets.

3.1.5 Implementation of Model Fitting and Overall Fairness Evaluation

The synthetic datasets, each identified by a unique `Dataset_ID`, were combined into a single consolidated dataset, as described in previous section. From this combined file, each synthetic dataset was isolated by filtering on its `Dataset_ID` to enable independent model training and fairness evaluation. For each of the 10,000 synthetic datasets, a least squares regression model was trained independently using the `LinearRegression` class from the `scikit-learn` library, from which the learned model coefficients θ_0 (intercept) and θ_1 (slope) were extracted.

Following model training, the discrimination statistics $d(\theta)$ for each of the 10,000 datasets were computed using the function defined above. The results, including the dataset identifier, discrimination statistic, and model coefficients, were compiled into a single CSV file, enabling subsequent statistical analysis over all datasets.

One primary application of this consolidated CSV file was to calculate the mean discrimination statistic across all 10,000 datasets. This aggregate measure offers a comprehensive assessment of the overall bias exhibited by the fitted models.

The computed mean discrimination statistic was approximately -0.666 , indicating a substantial level of bias favoring group B. Specifically, the negative value reveals that the model tends to under-predict the aptitude for individuals in group A (type 0) relative to those in group B (type 1), highlighting significant unfairness in the linear regression estimates across the synthetic datasets.

Given that the discrimination statistic reported in the original paper was approximately -0.67 , obtaining a similar value in this implementation served as a crucial verification step. It confirmed that the recreation of the example up to this point was performed correctly and that the synthetic data, model fitting, and fairness evaluation aligned with the original study's findings.

Moreover, this outcome provided strong empirical support for the paper's assertion that the use of ordinary least squares linear regression in this context can produce models exhibiting substantial discriminatory behavior, systematically favoring one group over another in their predictions. This highlights the inherent risks of bias in commonly used statistical methods when fairness considerations are not explicitly addressed.

3.1.6 Visualization of Discrimination in Linear Regression Predictions

To visually assess the presence of systematic bias in the fitted linear models, Figure S2 from the supplementary paper was recreated. This figure overlays 10,000 linear estimators obtained via least squares regression as transparent lines, providing insight into the variability and overall trend of the predictions across all synthetic datasets.

In the background, 5,000 samples from the synthetic data distribution (X, Y, T) are plotted to show the true underlying structure of the data. Each data point is colored according to group membership: red for individuals of type $T = 1$ and blue for type $T = 0$. The transparency of the lines helps visualize the density and spread of the learned models.

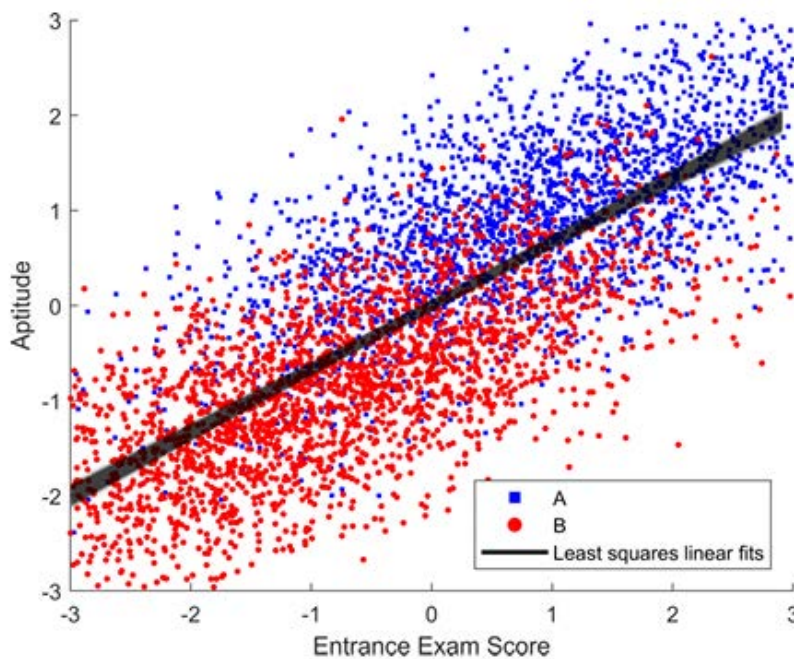


Figure 3.1: Figure S2 as presented in the paper

The recreated plot reveals a consistent pattern of overestimation for the red group ($T = 1$) and underestimation for the blue group ($T = 0$), illustrating the systematic discrimination captured by the average discrimination statistic. This visual confirmation complements the earlier numerical analysis, reinforcing the conclusion that linear regression, even without access to the sensitive attribute T , can encode bias present in the data distribution.

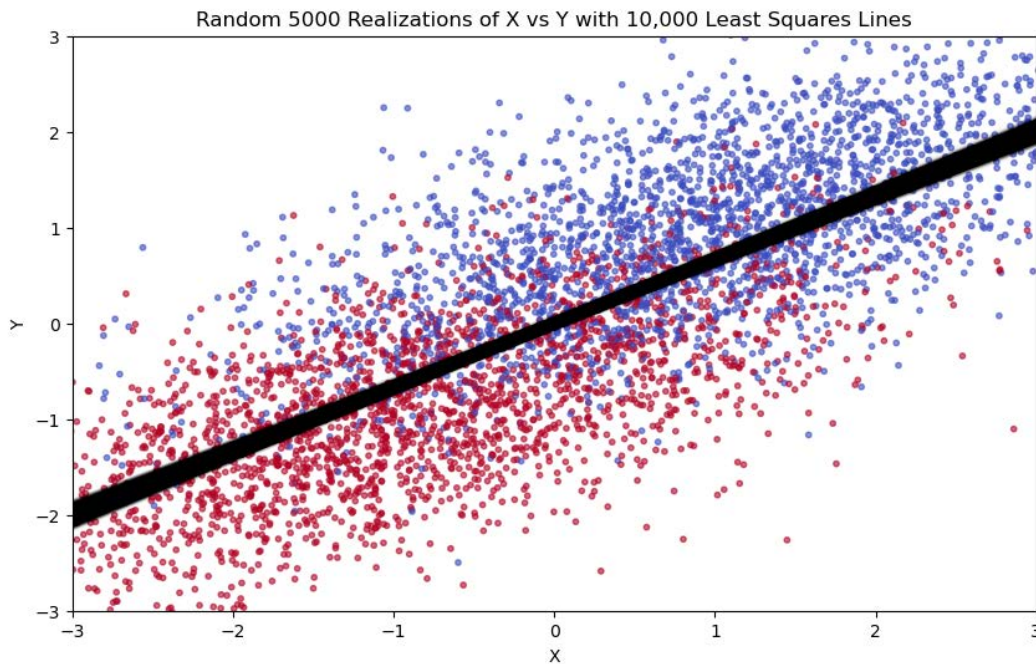


Figure 3.2: Recreation of Figure S2

3.2 Root causes and Potential Remedies for Undesirable Behavior

Having established the fairness issue through the recreated illustrative example, it becomes evident that optimizing solely for the primary objective can lead to undesirable discriminatory outcomes. This motivated the authors to explore of potential remedies—methods designed to mitigate or eliminate unfair behavior in machine learning models.

3.2.1 Understanding Root Causes of Undesirable Behavior

Before examining these remedies, the authors first sought to understand the root causes of such undesirable behavior. According to the paper, potential causes include improper algorithm use, selection of an unsuitable algorithm, imbalances or biases in the training data, limited or finite data samples, or inherent conflicts between the algorithm's design objectives—such as minimizing mean squared error (MSE)—and fairness goals. Identifying the responsible factor is challenging, particularly for users without deep expertise in machine learning or data analysis.

In analyzing the example, the authors ruled out common explanations. They found that the

discriminatory behavior was not due to biased or imbalanced training data, since the applicant groups were balanced and both training and testing data came from the same distribution. It was also not caused by the finite size of the training set, as even with infinite data, the linear least squares estimator remained biased. Furthermore, ignoring sensitive attributes like applicant type did not prevent discrimination, and training separate models for each group was impractical without access to those attributes and sufficient data. Ultimately, the root cause was identified as a conflict within the objective function itself: minimizing MSE clashes with minimizing discrimination. As a result, the estimator that optimizes MSE systematically favors one group over another, leading to unfair outcomes despite the simplicity and apparent fairness of the dataset and model.

Having identified this root cause, the original paper proposes several remedies and critically evaluates them. While these approaches aim to address fairness, each has its own limitations, which the authors discuss in detail.

In this section, I begin by summarizing the potential remedies proposed in the paper, along with their limitations and the reasons why none fully resolve fairness challenges or serve as ideal solutions in every case. This overview highlights both the promise and shortcomings of each method. Following that, I focus specifically on the remedy associated with the figure I recreated (Figure S4).

3.2.2 Potential Remedies

The potential remedies can be broadly categorized into the following:

- **Hard Constraints:** These enforce strict rules the model must obey, regardless of their impact on performance. For example, one might specify that no subgroup should receive systematically worse predictions. However, defining such constraints often requires detailed knowledge about the data distribution or the domain, which may not be feasible in practice.
- **Soft Constraints:** Rather than enforcing fairness absolutely, soft constraints introduce a penalty into the loss function to discourage undesirable behavior (e.g., discrimination). A key challenge with this method is selecting the right trade-off parameter, which balances predictive accuracy against fairness. In my experiments, I explored how varying this parameter affected model behavior (See Section 2.3 for the experimental anal-

ysis of soft constraints.).

- **Multiobjective Optimization:** This approach treats fairness and accuracy as separate but competing goals, and seeks solutions that strike a balance between them. Instead of collapsing these goals into a single scalar objective, multiobjective methods produce a range of solutions (a Pareto frontier), from which users can choose based on their preferences. However, even this approach does not fully eliminate the need to make trade-offs.
- **Chance-Constrained Optimization:** Here, constraints are allowed to be violated occasionally, but only with low probability. This is particularly useful when it's impossible to guarantee fairness with certainty due to noise or limited data. However, it often relies on knowing the distribution of the data-generating process, which may be unavailable in real-world scenarios.
- **Robust and Risk-Sensitive Methods:** These aim to guard against worst-case scenarios by focusing on solutions that perform reasonably well even under uncertainty. While these methods can improve reliability and reduce harmful behavior in edge cases, they generally don't allow fine-grained control over what constitutes undesirable behavior, such as discrimination.

Each of these remedies addresses a different aspect of the challenge, and their suitability depends on the specific application, available data, and how clearly undesirable behavior can be defined. In the following section, I will examine in more detail how soft constraints and their associated parameters impact model fairness and accuracy, using the experimental results from my reproductions of Figure S4 from the supplementary paper.

3.2.3 Experimental Results: Soft Constraints and Discrimination Mitigation

As seen before, one of the methods explored by the original authors to mitigate unfair outcomes involves adjusting the learning objective itself so that it takes fairness into account during training. Instead of strictly minimizing prediction error, the idea is to gently discourage discriminatory behavior by adding a penalty term to the loss function. This type of approach is often referred to as a soft constraint — it introduces a fairness-related signal to guide the

model, but doesn't strictly enforce it. This is done by adding a term to the loss function that captures the degree of discrimination (for example, the difference in prediction errors across groups). A tuning parameter, denoted λ , controls how much importance is assigned to this fairness penalty relative to prediction accuracy. As λ increases, the model gives more weight to reducing discrimination, which typically results in fairer outcomes—but often at the expense of higher error. Conversely, if λ is too small, the model remains focused on accuracy and continues to produce biased results. Choosing an appropriate λ is not trivial and generally requires careful tuning and extensive data analysis, which may not be feasible for typical users.

To explore how this method behaves in practice, the researchers evaluated its impact using the same illustrative example from earlier in the paper. They visualized how different values of λ affect both prediction error and the level of discrimination. As shown in the figure below (Figure S4 from the supplementary material), the authors observed that in order to achieve a mean discrimination statistic close to -0.1 , the optimal value of λ is approximately 4. For λ values less than 4, the discrimination statistic ranges from -0.67 (when $\lambda = 0$) up to -0.1 . In contrast, for λ values greater than 4, the discrimination statistic tends to approach 0, indicating reduced disparity across groups. However, this reduction in disparity comes at the cost of increased prediction error, as the mean squared error (MSE) approaches 1.

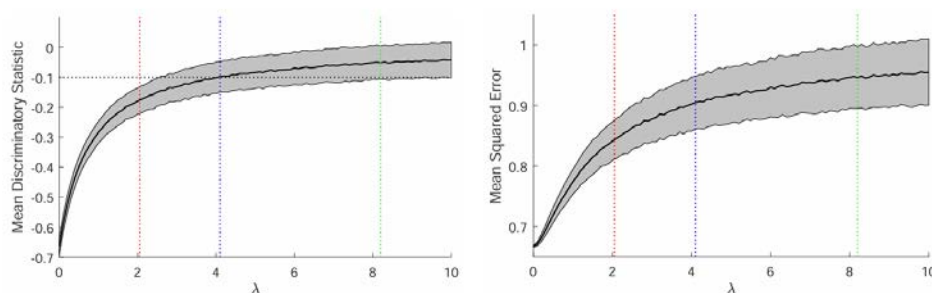


Figure 3.3: Figure S4 as presented in [1]

I recreated this figure based on their methodology in order to replicate their findings and gain insight into how effective this approach really is.

The following section presents my version of Figure S4, along with a discussion of what the results reveal about the trade-offs involved in using soft constraints to address fairness.

3.2.4 Recreation of Soft constraint experiment

Since the illustrative example is used, the model is a simple linear regression:

$$\hat{Y} = \theta_1 X + \theta_2$$

where X is the entrance exam score, Y is the aptitude and (θ_1, θ_2) are the model parameters.

To incorporate fairness, the following composite objective is defined:

$$f(\theta) = \text{MSE}(\theta) + \lambda \cdot |d(\theta)|$$

Here:

- $\text{MSE}(\theta) = \mathbb{E} [(\hat{Y} - Y)^2]$ is the mean squared error,
- $d(\theta) := \mathbb{E} [u(y(X, \theta) - Y) \mid T = 0] - \mathbb{E} [u(y(X, \theta) - Y) \mid T = 1]$ is the **discrimination statistic**, representing residual disparities across the sensitive attribute $T \in \{0, 1\}$,
- λ is a non-negative regularization parameter controlling the trade-off between fairness and accuracy.

To explore the fairness-accuracy trade-off, I performed the following steps:

1. **Sampling:** For each trial, I randomly sampled a subset of data points from the dataset.
2. **Optimization:** For each value of λ in a pre-defined range $([0, 10])$, I minimized the objective function $f(\theta)$ using gradient-based optimization.
3. **Evaluation:** After training, I computed:
 - the mean squared error (MSE),
 - the discrimination statistic $d(\theta)$,

for the learned parameters θ .

4. **Averaging:** This process is repeated over multiple random trials. I computed the average MSE and discrimination statistic across trials for each λ .

This approach yields two primary diagnostic plots:

- **Discrimination vs. λ :** quantifies the fairness improvement.

- **MSE vs. λ :** quantifies the accuracy degradation.

To match the original figure, I additionally identified a threshold value of λ at which the discrimination statistic falls below a pre-defined fairness cutoff. In the original experiment, the authors set this threshold at $|d(\theta)| < 0.1$. However, in my implementation, I relaxed this condition slightly to $|d(\theta)| < 0.15$, as the discrimination statistic was converging toward 0.1 very slowly. To better understand the behavior of the model around this critical point, I also examined the values $\lambda/2$ and 2λ to assess sensitivity to the fairness–accuracy trade-off.

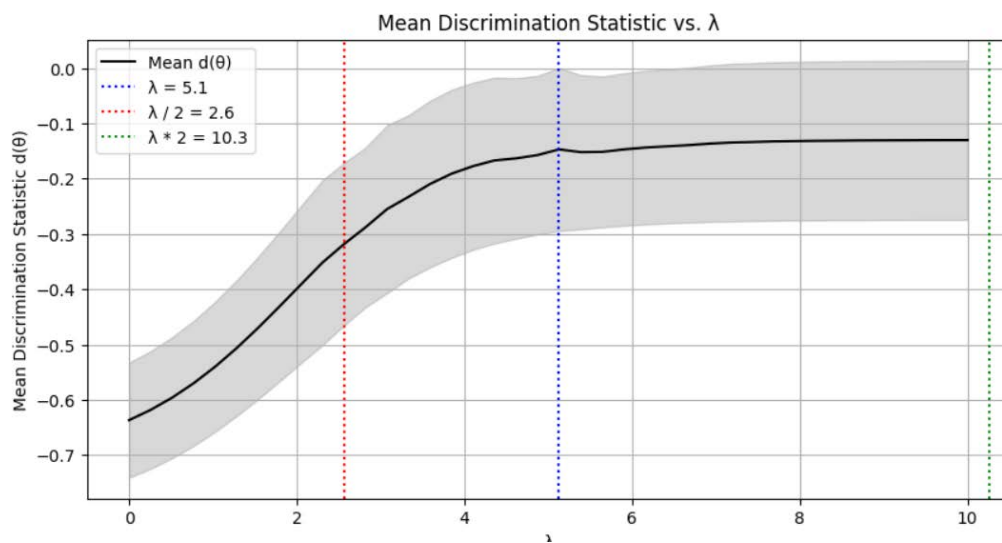


Figure 3.4: Discrimination vs. λ

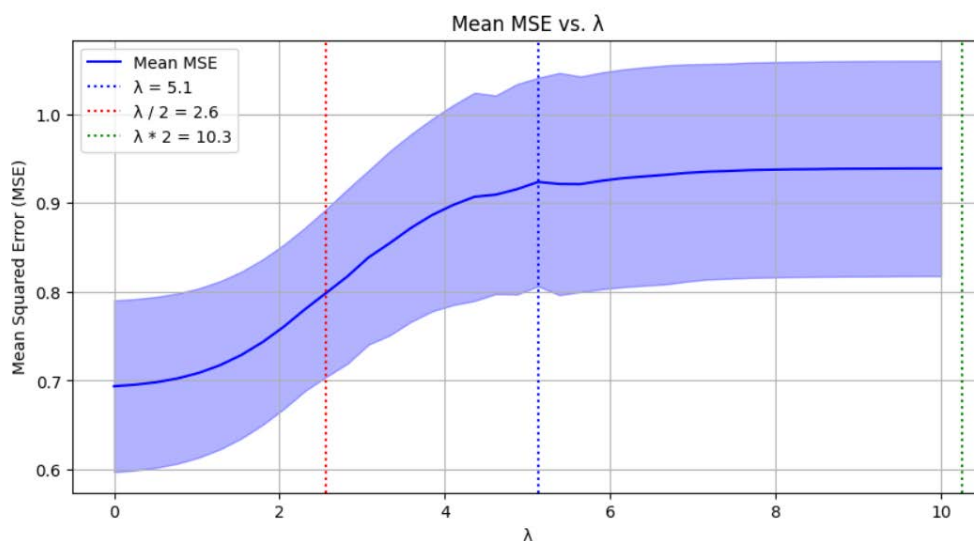


Figure 3.5: MSE vs. λ

Analysis and Interpretation of Results

From the resulting plots, a clear trade-off between fairness and accuracy emerges as λ increases. At $\lambda = 0$, where no fairness constraint is applied, the model achieves the lowest mean squared error (MSE), but exhibits high discrimination, with the discrimination statistic $d(\theta)$ reaching values as extreme as -0.67 . As λ increases, the discrimination statistic progressively moves closer to zero, indicating improved fairness across the sensitive attribute. However, this comes at the cost of increased MSE, especially beyond $\lambda \approx 4$, where fairness gains plateau but predictive performance continues to degrade. Around the threshold value of $\lambda \approx 5.1$ —where $|d(\theta)|$ drops below 0.15 —the trade-off appears most balanced: fairness is significantly improved while accuracy remains relatively acceptable. Exploring $\lambda/2$ and 2λ around this critical point further highlights how sensitive the model is to small changes in regularization strength. These results reinforce the importance of careful λ selection depending on application-specific tolerance for fairness versus accuracy.

These experimental results further reinforce the nuanced nature of employing soft constraints in fairness-aware learning. While it is encouraging to observe that appropriate values of λ can indeed lead to reduced discrimination without severely compromising accuracy, the process of identifying such a value is highly sensitive and data-dependent. The gradual decline of the discrimination statistic and the non-linear behavior of MSE across λ suggest that there is no universally optimal regularization weight. Instead, effective tuning requires iterative experimentation and careful observation. Moreover, my decision to relax the discrimination threshold to $|d(\theta)| < 0.15$ further highlights this point: without such an adjustment, it was difficult to reach the original fairness threshold of 0.1 within a reasonable computational budget or learning rate. This underscores the empirical reality that soft fairness constraints, while flexible, require substantial experimentation and calibration to be effective in practice.

What stands out is the delicate balance required to navigate between fairness and performance. Even slight deviations around the optimal λ point (e.g., testing $\lambda/2$ and 2λ) demonstrate how rapidly the trade-off shifts, which emphasizes that while soft constraints are promising, they do not offer a plug-and-play solution. These findings lend additional credibility to the authors' original argument: soft constraints are a flexible tool for fairness, but their practical effectiveness hinges on context-specific adjustments. Ultimately, this underscores the need for fairness-aware model tuning to be approached not just algorithmically, but also with domain insight and critical evaluation.

Code Implementation and Technical details

This subsection outlines the computational implementation of the objective function and optimization procedure used to recreate Figure S4. Two versions of the implementation were explored: an initial custom gradient descent method built from scratch using NumPy, and a final version utilizing TensorFlow’s automatic differentiation and the Adam optimizer. Each approach involved minimizing a fairness-regularized loss function that balances predictive accuracy with group fairness.

Initial Approach: Custom Gradient Descent using NumPy. The first implementation of the fairness-aware optimization relied on a custom gradient descent algorithm built entirely with NumPy. This version aimed to recreate the fairness-regularized objective from the original paper using manual gradient calculations, with the goal of minimizing prediction error while incorporating fairness via a regularization term. Despite initial efforts, this approach was ultimately unsuccessful due to several practical limitations.

In implementing the optimization manually using NumPy, a crucial modification was made to the original objective function. Specifically, although the theoretical fairness penalty was defined using the absolute value of the discrimination statistic,

$$f(\theta) = \text{MSE}(\theta) + \lambda \cdot |d(\theta)|,$$

this formulation is not fully differentiable at $d(\theta) = 0$, which poses challenges for gradient-based optimization. Since NumPy does not support automatic differentiation, and subgradients are cumbersome to implement correctly in this context, I replaced the absolute value with its squared form:

$$f(\theta) = \text{MSE}(\theta) + \lambda \cdot d(\theta)^2.$$

This ensures that the entire objective is smooth and differentiable, allowing for the use of manually computed gradients. While this change slightly alters the optimization landscape, it retains the core intent: penalizing large disparities in residuals across sensitive groups.

To minimize this loss, I implemented a manual gradient descent routine. The gradients of both the MSE and the fairness term were computed analytically with respect to the model parameters θ_1 and θ_2 . A learning rate and gradient clipping were applied to stabilize updates. Additionally, I introduced an adaptive learning rate of the form $\frac{1}{1+\lambda}$ to handle numerical sensitivity for larger λ values.

Despite careful construction, this approach struggled to reliably reduce the discrimination statistic to the desired threshold ($|d(\theta)| < 0.15$). In particular, the manually computed gradients for the fairness component introduced instability. These issues motivated the transition to a more robust optimization framework.

Final Implementation: TensorFlow Optimization with Adam

The second, successful implementation leveraged TensorFlow’s automatic differentiation and symbolic computation capabilities, combined with the Adam optimizer, to yield a robust and stable training process. This approach offered significant improvements over the earlier NumPy-based method, particularly in terms of gradient handling, numerical stability, and convergence behavior.

The model architecture remained simple, employing a linear regression model:

$$\hat{Y} = \theta_1 X + \theta_2$$

where both θ_1 and θ_2 were represented as `tf.Variable` tensors of type `float32`, allowing TensorFlow to automatically track gradients during optimization.

Loss Function and Fairness Penalty. The total objective function was defined as:

$$\mathcal{L}(\theta) = \text{MSE}(\theta) + \tilde{\lambda} \cdot |d(\theta)|$$

with a smoothed penalty weight:

$$\tilde{\lambda} = \frac{1}{1 + e^{-(\lambda-2)}}$$

This sigmoid-scaling helped prevent steep gradients when λ was large, thus avoiding premature convergence or instability early in training. The mean squared error was computed as:

$$\text{MSE}(\theta) = \mathbb{E}[(\hat{Y} - Y)^2]$$

The discrimination statistic $d(\theta)$ was computed using TensorFlow operations that separated the residuals by sensitive group. Specifically, the residuals $r = \hat{Y} - Y$ were computed for each datapoint, and then conditionally averaged using `tf.where(T == 0)` and `tf.where(T == 1)` to isolate the expected residuals for each group. The difference between these conditional expectations yields the fairness penalty. Gradient computation was handled using

TensorFlow’s `GradientTape`, which efficiently tracked all operations and allowed automatic differentiation of the composite objective, even with non-standard loss components. For numerical stability, gradients were clipped within a fixed range (± 10). The key block for loss evaluation and gradient tracking was structured as follows:

```
with tf.GradientTape(persistent=True) as tape:
    tape.watch(θ1)
    tape.watch(θ2)

    λ = tf.cast(λ, tf.float32)
    scaled_lambda = 1 / (1 + tf.exp(-λ + 2))

    loss_value = mse(X, Y, θ1, θ2) + scaled_lambda * tf.abs(discrimination_statistic(X, Y, T, θ1, θ2))
```

Figure 3.6: Gradient computation was handled using TensorFlow’s `GradientTape`

Optimization Strategy. The optimization was carried out using the Adam optimizer with carefully chosen hyperparameters. The parameters θ_1 and θ_2 were initialized with small Gaussian noise (standard deviation ≈ 0.01) to break symmetry. The optimization loop used 100 iterations per run, with a fixed learning rate of 0.007. The Adam optimizer was configured with:

$$\beta_1 = 0.70, \quad \beta_2 = 0.6, \quad \epsilon = 10^{-8}$$

These relatively low momentum terms helped the optimizer better adapt to the noisy gradient signals from the fairness term, which can be highly variable depending on group sample sizes.

Gradient clipping was also applied to both parameters:

$$\text{grad}_\theta \leftarrow \text{clip}(\text{grad}_\theta, -10, 10)$$

to prevent large updates from destabilizing the parameter space.

Experimental Design. For robustness, the model was trained across 10 independent trials, each using a random sample of 100 data points from the synthetic dataset. The procedure was repeated across 40 values of $\lambda \in [0, 10]$, equally spaced. For each λ and trial, we stored:

- The MSE of the trained model,
- The absolute value of the discrimination statistic $|d(\theta)|$.

After all runs, the mean and standard deviation of both metrics were computed across trials. The fairness threshold was then identified as the smallest λ for which the averaged discrimination statistic satisfied:

$$|d(\theta)| < 0.15$$

This relaxed criterion reflects the computational cost of tighter thresholds.

Outcome. This implementation successfully reproduced the shape and trade-off patterns observed in the original Figure S4 of the supplementary material.

Chapter 4

Methodology - The Seldonian Framework

This chapter presents the solution proposed by the original authors to the fairness problem demonstrated in the previous chapter. It begins by introducing the theoretical foundations of the Seldonian framework and formalizing the Seldonian optimization problem, which enables the enforcement of user-defined fairness constraints with high confidence. The chapter then provides a detailed decomposition of the two core algorithms—NDLR and QNDLR—translating their pseudocode into functional Python implementations. Finally, it outlines the design of the experimental setups developed to evaluate these algorithms, including both synthetic-data experiments and comparative analyses against state-of-the-art fairness-aware regression techniques.

4.1 Towards Fairer Models: The Seldonian Approach

Having examined the inherent fairness issues present in standard machine learning pipelines particularly those involving linear regression—and having evaluated the shortcomings of conventional bias mitigation strategies, it becomes clear that a more principled framework is needed. Despite the increasing awareness of fairness constraints in algorithmic decision-making, many existing techniques either offer no formal guarantees or fail to generalize beyond specific contexts or datasets.

In this chapter i will present the authors' solution to this existing problem - The seldonian solution.

4.2 The Seldonian Framework

This framework, proposed by the authors of “Preventing undesirable behavior of intelligent machines” [1] paper, addresses limitations in standard machine learning (ML) formulations particularly the difficulty of incorporating user-specified behavioral constraints without requiring detailed domain knowledge. Inspired by Isaac Asimov’s fictional character Hari Seldon, the framework introduces a formal mechanism for ensuring that ML algorithms produce safe, fair, or otherwise acceptable behavior with high probability.

4.2.1 Motivation and Limitations of Prior Approaches

In standard ML practice, algorithms are typically designed to optimize a performance objective (e.g., prediction accuracy or mean squared error) without accounting for potential undesirable outcomes such as discrimination or unsafe decisions. Some might attempt to address this by placing probabilistic constraints on the final model’s behavior. However, defining safety as a property of a particular model often leads to degenerate formulations where constraints become trivial (probability 0 or 1), or infeasible, because the true safe set of models is unknown beforehand.

Another attempted workaround involves encoding safety in terms of the algorithm’s belief about the model’s behavior based on data. This belief-driven approach leads to a different issue: the feasible set of solutions becomes data-dependent and random, undermining its ability to reflect stable design objectives. Furthermore, it fails to prevent unsafe outcomes with high probability, particularly when multiple constraints are present.

4.2.2 Seldonian Optimization Problem (SOP)

To overcome these issues, the authors propose a fundamental reformulation: shifting the focus from optimizing over solutions to optimizing over algorithms. In this formulation, safety and other behavioral constraints are imposed not on individual models, but on the algorithms themselves, ensuring that the algorithm will—across random draws of data—rarely produce unsafe solutions.

Formally, let $\mathcal{A}$ denote the space of all candidate algorithms. Each algorithm $a \in \mathcal{A}$ is a function that maps a dataset D to a solution $\theta \in \Theta$. A Seldonian Optimization Problem (SOP) takes the form:

$$\arg \max_{a \in \mathcal{A}} f(a) \quad (4.1)$$

$$\text{s.t. } \Pr(g_i(a(D)) \leq 0) \geq 1 - \delta_i, \quad \forall i = 1, \dots, n \quad (4.2)$$

Here:

- $f(a)$ is the objective function defined over algorithms.
- Each g_i is a behavioral constraint function indicating the presence of undesirable behavior.
- $\delta_i \in [0, 1]$ represents a user-specified upper bound on the probability that constraint g_i is violated.
- D is a random dataset drawn from a distribution over possible training sets.

This formulation ensures that the probability an algorithm produces a model violating any given constraint is bounded by δ_i . The key distinction from previous approaches is that the Seldonian framework imposes behavioral constraints on the algorithm's output distribution—ensuring that, across many possible datasets, the probability of producing an unsafe or unfair solution remains below a user-specified threshold.

4.2.3 Interpretation and Practical Impact

The Seldonian framework provides guarantees reminiscent of the Probably Approximately Correct (PAC) learning framework [21]. The randomness in the constraint satisfaction arises from the randomness in the training dataset D , which in turn affects the solution $a(D)$ produced by the algorithm. Each behavioral constraint g_i is interpreted as a test of whether the resulting model causes undesirable behavior. By bounding the probability that $g_i(a(D)) > 0$, the algorithm designer ensures that only rarely does the algorithm produce harmful or unfair outcomes.

From the user's perspective, this approach offers significant advantages. Users are not required to understand the technical properties of specific models or estimators. Instead, they only need to specify what constitutes undesirable behavior and choose acceptable confidence levels δ_i . The burden of reasoning about which models satisfy those constraints is placed entirely on the algorithm, which performs the necessary internal analysis during training.

For instance, in a linear regression setting, a designer might define the objective function $f(a)$ as the negative mean squared error, and choose $\mathcal{A}$ to be the set of all affine functions. A user could then select a fairness constraint (such as limiting demographic bias) and an acceptable failure rate. The resulting Seldonian algorithm would ensure—with high probability—that the output model satisfies the specified fairness condition, regardless of the underlying data variability.

4.2.4 Conceptual Shift in Algorithm Design

This framework marks a conceptual shift in how ML algorithms are designed and applied. The complexity of enforcing safety and fairness is absorbed into the algorithm design process, thereby reducing the burden on end users. This is especially valuable in real-world applications where users may lack expertise in statistical reasoning or algorithmic behavior, yet must still ensure that outcomes are safe, fair, or legally compliant.

Although the design of Seldonian algorithms is more involved than traditional ML pipelines, the resulting systems offer stronger, interpretable guarantees. They are particularly suitable for high-stakes applications—such as healthcare, criminal justice, and finance—where even rare undesirable outcomes can have serious consequences.

4.3 Seldonian and Quasi Seldonian Algorithms

While the *Seldonian Optimization Problem (SOP)* provides a rigorous way to design machine learning algorithms that respect behavioral constraints, actually finding a solution to such a problem is often extremely difficult in practice. This is because solving the SOP exactly would require searching over all possible algorithms to find one that not only performs well (i.e., maximizes the objective function) but also guarantees—with high confidence—that undesirable behavior will not occur. This kind of optimization is typically intractable.

Because of this, the authors propose a more practical approach: instead of trying to find the optimal algorithm, the goal should at least be to find an algorithm that satisfies the behavioral constraints. Any algorithm that does this—whether it’s optimal or not—is called a Seldonian algorithm.

However, there’s still a problem. Verifying that an algorithm satisfies a constraint with high confidence usually requires strong statistical guarantees, often using tools like Hoeffd-

ing's inequality (see Chapter 2). These are known as concentration inequalities, and while they work for many types of data, they tend to be very conservative. As a result, a Seldonian algorithm might need a large amount of data just to be confident that it behaves safely. In some cases, that amount of data may not be available.

To address this issue, the authors introduce *quasi-Seldonian algorithms*. These algorithms relax the strict statistical guarantees by using approximate methods like Student's t-test or bootstrap confidence intervals. These methods often give much tighter (less conservative) results, which means they can be used effectively even with smaller datasets. However, they come with a tradeoff: they rely on certain assumptions (e.g., normality of the data), which might not always hold true. So the guarantees they provide are not as strong.

A key concern here is transparency: users of quasi-Seldonian algorithms may not always be aware of what assumptions the algorithm makes. If those assumptions are wrong, the guarantees of safety or fairness might not hold. Therefore, it's important for researchers to clearly state any assumptions their algorithm depends on.

Sometimes, even using approximate methods, it's not possible to find a solution that satisfies the constraints. In such cases, instead of forcing a potentially unsafe decision, the algorithm is allowed to return "No Solution Found" (NSF). This means the algorithm wasn't able to find a safe enough solution given the available data.

Of course, if NSF is an allowed option, then an algorithm that always returns NSF would technically satisfy the Seldonian definition—but that wouldn't be useful. To prevent this, the objective function in the SOP should give low utility to always returning NSF, so the algorithm is encouraged to only fall back on NSF when absolutely necessary.

In the next sections, I focus on how these general ideas are applied in practice through a specific case: Non-Discriminatory Linear Regression (NDLR). This algorithm is designed to perform standard linear regression while ensuring that the predictions do not discriminate based on sensitive attributes, such as race or gender. NDLR is a concrete example of a Seldonian algorithm, as it guarantees with high probability that a specified fairness constraint is satisfied. However, as discussed earlier, such guarantees often require large amounts of data. To address this, a relaxed version called quasi-NDLR (QNDLR) is introduced. QNDLR applies the same fairness constraint using approximate statistical bounds, enabling its use in lower-data settings at the cost of weaker guarantees. These algorithms demonstrate how the Seldonian framework can be instantiated to create fair, interpretable, and practical machine

learning models.

4.3.1 NDLR: Non Discriminatory Linear Regression

The *Non-Discriminatory Linear Regression (NDLR)* algorithm is a concrete example of a Seldonian algorithm applied to linear regression, where the goal is to ensure fairness with high confidence. Specifically, NDLR aims to find a linear model that not only performs well (i.e., minimizes prediction error), but also avoids discriminatory behavior with respect to a sensitive attribute (e.g., gender or race).

The algorithm proceeds in three main steps:

1. **Data Splitting:** The dataset D is partitioned into two disjoint subsets: D_1 and D_2 . The first set D_1 is used to select a *candidate solution* c , which is a model expected to satisfy the behavioral (fairness) constraint. The second set D_2 is reserved for an independent safety test.
2. **Candidate Selection:** Using D_1 , the algorithm searches for a linear regression model c that minimizes the mean squared error (MSE) while also being likely to pass the fairness test. To guide this, it uses the `HoeffdingCandidateObjective` function, which relies on Hoeffding's inequality to estimate whether a model will likely be accepted in the next phase.
3. **Safety Test (Fairness Check):** The selected candidate model c is tested on D_2 to verify whether it satisfies the fairness constraint. Specifically, the algorithm computes a high-confidence upper bound on the *absolute discrimination statistic* using the `HoeffdingDiscrimUpperBound` function. If this upper bound is less than a given threshold ϵ , the candidate is accepted and returned. Otherwise, the algorithm returns *No Solution Found*.

This structure ensures that with high probability $(1 - \delta)$, the returned model does not violate the fairness constraint. The use of disjoint datasets D_1 and D_2 is essential to maintain statistical validity by ensuring that the safety test is unbiased and does not reuse training data.

In summary, NDLR is designed to balance two objectives: minimizing prediction error and satisfying a fairness constraint under probabilistic guarantees. If a suitable model cannot be found with the available data, the algorithm refuses to return an unfair model by outputting *No Solution Found*.

Decomposition of the NDLR Pseudocode

1. HoeffdingUpperBound

This subroutine applies Hoeffding's inequality to compute a high-confidence upper bound on the mean of a bounded random variable. Let $Z = \{Z_1, Z_2, \dots, Z_m\}$ be a sample of m independent observations from a bounded random variable, and let $b \in \mathbb{R}_{\geq 0}$ be the known range width (i.e., the variable lies in some interval of length b).

Given a confidence parameter $\delta \in (0, 1)$, Hoeffding's inequality provides the following upper bound on the true expectation μ of the random variable:

$$\mu \leq \frac{1}{m} \sum_{i=1}^m Z_i + b \cdot \sqrt{\frac{\ln(1/\delta)}{2m}} \quad \text{with probability at least } 1 - \delta.$$

1 return $\frac{1}{m} (\sum_{i=1}^m Z_i) + b \sqrt{\frac{\ln(1/\delta)}{2m}};$

Figure 4.1: HoeffdingUpperBound(Z, b, δ) [1]

2. PredictHoeffding

The PredictHoeffding function is used to conservatively estimate what the Hoeffding upper bound would return if it were applied to a new dataset of k samples drawn from the same distribution. Over prediction is used to avoid underestimation, which could cause the NDLR algorithm to reject viable solutions unnecessarily.

Let $Z = \{Z_1, Z_2, \dots, Z_m\}$ be the current sample, $b \in \mathbb{R}_{\geq 0}$ the known range of the random variable, $\delta \in (0, 1)$ the confidence level, and $k \in \mathbb{N}$ the hypothetical number of future samples. Then the predicted upper bound is given by:

$$\text{PredictedUpperBound} = \frac{1}{m} \sum_{i=1}^m Z_i + b \cdot \sqrt{\frac{\ln(1/\delta)}{2k}}$$

1 return $\frac{1}{m} (\sum_{i=1}^m Z_i) + b \sqrt{\frac{\ln(1/\delta)}{2k}};$

Figure 4.2: PredictHoeffding(Z, b, δ, k)[1]

3. HoeffdingCandidateObjective

This function defines the objective minimized during candidate selection in the NDLR algorithm. It evaluates both prediction performance and a conservative estimate of discrimination. Given a dataset D with sensitive attribute labels, it computes the pairwise differences in prediction errors between the two groups and constructs a sample array Z . It then applies the `PredictHoeffding` function to estimate a high-confidence upper bound on the absolute discrimination statistic. If the estimated bound is below the discrimination threshold ϵ , the function returns the mean squared error (MSE) of predictions; otherwise, it returns a penalized value to deprioritize unfair solutions. This mechanism ensures that selected candidates are both accurate and likely to pass the safety test.

```

1 Input: 1) Solution  $\theta$ , 2) data set  $D = \{(X_i, Y_i, T_i)\}_{i=1}^m$ , 3) confidence level  $\delta \in (0, 1)$ ,
          4) maximum level of discrimination  $\epsilon \in \mathbb{R}_{>0}$ , 5) an upper bound  $b$  on the
          range of possible prediction errors, 6) a number of samples  $k$ .
2  $m_1 \leftarrow \sum_{i=1}^m T_i$ ,  $m_0 \leftarrow m - m_1$ ;
3 Let  $(X_i^0, Y_i^0)$  be the  $i^{\text{th}}$  data point of type zero, and  $(X_i^1, Y_i^1)$  be the  $i^{\text{th}}$  data point of
  type one;
4 Create array  $Z$  of length  $\min\{m_0, m_1\}$ , with values  $Z_i = (\theta^\top X_i^0 - Y_i^0) - (\theta^\top X_i^1 - Y_i^1)$ ;
5  $ub =$ 
   $\max\{\text{PredictHoeffding}(Z, b, \delta/2, k), \text{PredictHoeffding}(-Z, b, \delta/2, k)\};$ 
6 if  $ub \leq \epsilon$  then
7   return  $\frac{1}{m} \sum_{i=1}^m (\theta^\top X_i - Y_i)^2$ ;
8 return  $b^2 + ub - \epsilon$ ;

```

Figure 4.3: `HoeffdingCandidateObjective`($\theta, D, \delta, \epsilon, b, k$) [1]

4. HoeffdingDiscrimUpperBound

This function computes a high-confidence upper bound on the discrimination statistic of a given solution using Hoeffding's inequality. It computes the prediction pairwise error differences and stores the results in an array Z . To handle the absolute value in the discrimination constraint, the function computes two upper bounds using `HoeffdingUpperBound`, once for Z and once for $-Z$, and returns the maximum of the two. This ensures that the bound applies to the absolute discrimination statistic, maintaining the validity of the behavioral constraint with high probability.

```

1 Input: 1) Solution  $\theta$ , 2) data set  $D = \{(X_i, Y_i, T_i)\}_{i=1}^m$ , 3) confidence level  $\delta \in (0, 1)$ ,
          4) maximum level of discrimination  $\epsilon \in \mathbb{R}_{>0}$ , 5) an upper bound  $b$  on the
          range of possible prediction errors.
2  $m_1 \leftarrow \sum_{i=1}^m T_i$ ,  $m_0 \leftarrow m - m_1$ ;
3 Let  $(X_i^0, Y_i^0)$  be the  $i^{\text{th}}$  data point of type zero, and  $(X_i^1, Y_i^1)$  be the  $i^{\text{th}}$  data point of
  type one;
4 Create array  $Z$  of length  $\min\{m_0, m_1\}$ , with values  $Z_i = (\theta^\top X_i^0 - Y_i^0) - (\theta^\top X_i^1 - Y_i^1)$ ;
5 return
   $\max\{\text{HoeffdingUpperBound}(Z, b, \delta/2), \text{HoeffdingUpperBound}(-Z, b, \delta/2)\}$ ;

```

Figure 4.4: HoeffdingDiscrimUpperBound($\theta, D, \delta, \epsilon, b$)[1]

5. NDLR

The final function implements the Non-Discriminatory Linear Regression (NDLR) algorithm. It takes as input a dataset $D = \{(X_i, Y_i, T_i)\}_{i=1}^m$, a confidence level $\delta \in (0, 1)$, a discrimination threshold $\epsilon > 0$, and an upper bound b on the prediction error range. The dataset is partitioned into D_1 (20%) and D_2 (80%). A candidate solution θ_c is selected by minimizing the HoeffdingCandidateObjective using D_1 , while accounting for the discrimination constraint to be tested on D_2 . The algorithm then evaluates whether θ_c satisfies the discrimination constraint using HoeffdingDiscrimUpperBound. If the upper bound is within ϵ , the solution is returned; otherwise, no solution is returned.

```

1 Input: 1) Data set  $D = \{(X_i, Y_i, T_i)\}_{i=1}^m$ , 2) confidence level  $\delta \in (0, 1)$ , 3) maximum
          level of discrimination  $\epsilon \in \mathbb{R}_{>0}$ , 4) an upper bound  $b$  on the magnitude of a
          prediction error.
2 Partition  $D$  into  $D_1$  (20% of data) and  $D_2$  (80% of data);
3  $\theta_c \in \arg \min_{\theta \in \Theta} \text{HoeffdingCandidateObjective}(\theta, D_1, \delta, \epsilon, b, |D_2|)$ ;
4 if  $\text{HoeffdingDiscrimUpperBound}(\theta_c, D_2, \delta, \epsilon, b) \leq \epsilon$  then
5   return  $\theta_c$ ;
6 return NO SOLUTION FOUND;

```

Figure 4.5: Non-Discriminatory Linear Regression (NDLR) [1]

Python Implementation of NDLR: Technical Highlights

The following are key technical aspects of my Python implementation of the NDLR algorithm:

- **Random partitioning:** I used `np.random.shuffle` to randomly split the dataset into D_1 and D_2 , preserving statistical independence between candidate selection and safety testing.

- **Vectorized operations:** All error and prediction calculations (e.g., computing Z) were implemented using NumPy vectorization for efficiency and clarity.
- **Optimization routine:** For parameter selection,

I used the `scipy.optimize.minimize` function with the Nelder–Mead simplex method. While standard gradient-based methods such as BFGS or Conjugate Gradient (CG) are theoretically faster and commonly preferred when gradients are available, they rely on smooth, differentiable objective functions. In my implementation, the objective function (see Fig. 4.5) due to its dependence on Hoeffding-based upper bounds and conditional penalties—exhibited non-smooth and potentially noisy behavior.

When I initially tested gradient-based optimizers, I encountered either extremely slow convergence or failure to reach a solution within the iteration limits. These issues indicated that the objective landscape was likely ill-conditioned or discontinuous, making it poorly suited for gradient-driven methods.

In contrast, the Nelder–Mead algorithm, which does not require derivative information, proved far more robust in this context. Despite its relatively slower convergence in general, it performed reliably and converged effectively for this low-dimensional, non-smooth optimization task. This made it the most suitable and stable choice for my setting.

4.3.2 QNDLR(λ): Quasi Non Discriminatory Linear Regression

The Quasi-Non-Discriminatory Linear Regression (QNDLR) algorithm is a quasi-Seldonian variant of NDLR that retains the same overall structure and fairness objective but replaces the statistical machinery used to estimate high-confidence bounds. QNDLR is designed to be less conservative and more data-efficient than NDLR by leveraging parametric assumptions about the data distribution.

The algorithm proceeds in the same three phases as NDLR—data splitting, candidate selection, and safety testing—but with the following key differences:

- **Statistical Estimation:** Instead of Hoeffding’s inequality, QNDLR uses Student’s t -test to compute high-confidence upper bounds on the discrimination statistic. This sub-

stitution assumes that the empirical discrimination measurements are approximately normally distributed, which allows for tighter and less conservative bounds.

- **Discrimination Bound Functions:** The functions `TTestUpperBound` and `TTestDiscrimUpperBound` replace `HoeffdingUpperBound` and `HoeffdingDiscrimUpperBound`, respectively.
- **Candidate Objective Function:** In the candidate selection step, the algorithm optimizes `TTestCandidateObjective` instead of `HoeffdingCandidateObjective`.
- **Safety Guarantees:** Despite using parametric assumptions, QNDLR still offers probabilistic fairness guarantees: if it returns a solution, that model satisfies the discrimination constraint with high confidence (i.e., probability at least $1 - \delta$).
- **Soft-Constrained Variant (QNDLR(λ)):** A key distinction from NDLR is the existence of a soft-constrained version, denoted QNDLR(λ), which incorporates a tunable hyperparameter $\lambda \geq 0$ into the candidate objective. This version treats the fairness constraint not as a hard cutoff during candidate selection, but instead adds a penalty term proportional to the predicted discrimination upper bound. The objective becomes:

$$\text{MSE}(\theta) + \lambda \cdot \text{PredictedDiscriminationUpperBound}(\theta)$$

This modification turns the problem into a multi-objective optimization, where the trade-off between accuracy and fairness can be controlled. Crucially, despite softening the constraint, QNDLR(λ) still performs a final high-confidence fairness test using t -test bounds. If the candidate fails this test, it is rejected—thereby preserving the probabilistic fairness guarantees.

- **Computational Considerations:** Due to the use of t -test bounds and the more complex objective, QNDLR may be computationally more intensive than NDLR, but often benefits from greater solution availability in small- to medium-sized datasets where Hoeffding bounds are overly conservative.

Decomposition of the QNDLR Pseudocode

1. TTestUpperBound

This function computes an upper confidence bound on the mean of a random variable using Student's t -test. Given a sample vector $Z = [Z_1, \dots, Z_m]$, it first calculates the sample mean and unbiased sample variance. It then returns the upper bound as:

$$\bar{Z} + t_{1-\delta, m-1} \cdot \frac{\sigma}{\sqrt{m}}$$

where δ is the confidence level, and $t_{1-\delta, m-1}$ is the critical value from the t -distribution with $m - 1$ degrees of freedom. This conservative bound estimates the highest plausible true mean consistent with the observed data.

```

1  $\bar{Z} \leftarrow \frac{1}{m} \sum_{i=1}^m Z_i, \quad \sigma \leftarrow \sqrt{\frac{1}{m-1} \sum_{i=1}^m (Z_i - \bar{Z})^2};$ 
2 return  $\bar{Z} + \frac{\sigma}{\sqrt{m}} t_{1-\delta, m-1};$ 

```

Figure 4.6: TTestUpperBound(Z, δ) [1]

2. PredictTTest

This function predicts the value that TTestUpperBound(Z, δ) would return if applied to a future sample $Z_k = [Z_1, \dots, Z_k]$ of size k , introducing a deliberate overestimation to ensure safety under uncertainty. It first computes the sample mean and unbiased variance, then returns:

$$\bar{Z} + 2 \cdot t_{1-\delta, k-1} \cdot \frac{\sigma}{\sqrt{k}}$$

By scaling the confidence interval by a factor of 2, the function conservatively over-approximates the true bound. This is critical in the QNDLR framework, where underestimation can lead to frequent returning of NoSolutionFound.

```

1  $\bar{Z} \leftarrow \frac{1}{m} \sum_{i=1}^m Z_i, \quad \sigma \leftarrow \sqrt{\frac{1}{m-1} \sum_{i=1}^m (Z_i - \bar{Z})^2};$ 
2 return  $\bar{Z} + 2 \frac{\sigma}{\sqrt{k}} t_{1-\delta, k-1};$ 

```

Figure 4.7: PredictTTest(Z, b, δ, k) [1]

3. TTestCandidateObjective

The function TTestCandidateObjective defines the objective minimized during candidate selection in QNDLR (λ). It operates similarly to

HoeffdingCandidateObjective, with the key difference being the use of PredictTTest to estimate a high-confidence upper bound on the absolute discrimination statistic, instead of PredictHoeffding. The return value depends on whether the upper bound satisfies the fairness constraint. The function behaves as follows:

$$\text{TTestCandidateObjective}(\theta) = \begin{cases} \frac{1}{m} \sum_{i=1}^m (X_i \cdot \theta - Y_i)^2 + \lambda \cdot \frac{1}{|Z|} \sum_{i=1}^{|Z|} |Z_i|, & \text{if } \text{UB} \leq \epsilon, \\ b^2 + \text{UB} + (\lambda - 1)\epsilon, & \text{otherwise,} \end{cases}$$

where $\text{UB} = \max(\text{PredictTTest}(Z, b, \delta/2, k), \text{PredictTTest}(-Z, b, \delta/2, k))$.

When $\lambda = 0$, the objective reduces to minimizing prediction error alone, ignoring the fairness term. The fairness constraint $\text{UB} \leq \epsilon$ is still enforced, but among the fair candidates, the one with the lowest mean squared error is preferred.

```

1 Input: 1) Solution  $\theta$ , 2) data set  $D = \{(X_i, Y_i, T_i)\}_{i=1}^m$ , 3) confidence level  $\delta \in (0, 1)$ ,
          4) maximum level of discrimination  $\epsilon \in \mathbb{R}_{>0}$ , 5) a number of samples  $k$ , 6) a
          constant  $\lambda \in \mathbb{R}_{\geq 0}$  that balances the trade-off between MSE and
          discrimination.
2  $m_1 \leftarrow \sum_{i=1}^m T_i$ ,  $m_0 \leftarrow m - m_1$ ;
3 Let  $(X_i^0, Y_i^0)$  be the  $i^{\text{th}}$  data point of type zero, and  $(X_i^1, Y_i^1)$  be the  $i^{\text{th}}$  data point of
  type one;
4 Create array  $Z$  of length  $\min\{m_0, m_1\}$ , with values  $Z_i = (\theta^\top X_i^0 - Y_i^0) - (\theta^\top X_i^1 - Y_i^1)$ ;
5  $\text{ub} = \max\{\text{PredictTTest}(Z, b, \delta/2, k), \text{PredictTTest}(-Z, b, \delta/2, k)\}$ ;
6 if  $\text{ub} \leq \epsilon$  then
7   return  $\frac{1}{m} \sum_{i=1}^m (\theta^\top X_i - Y_i)^2 + \lambda \frac{1}{|Z|} \sum_{i=1}^{|Z|} |Z_i|$ ;
8 return  $b^2 + \text{ub} + (\lambda - 1)\epsilon$ ;

```

Figure 4.8: TTestCandidateObjective($\theta, D, \delta, \epsilon, k, \lambda$) [1]

4. TTestDiscrimUpperBound

This function mirrors the logic of HoeffdingDiscrimUpperBound, but replaces the Hoeffding bound with TTestUpperBound. It computes residual differences Z_i between group 0 and group 1, and then applies the one-sided t -test upper bound to both Z and $-Z$ to conservatively estimate group-level discrimination. Unlike the Hoeffding-based version, which assumes no distributional information, the t -test variant assumes approximate normality and uses sample variance.

```

1 Input: 1) Solution  $\theta$ , 2) data set  $D = \{(X_i, Y_i, T_i)\}_{i=1}^m$ , 3) confidence level  $\delta \in (0, 1)$ ,
           4) maximum level of discrimination  $\epsilon \in \mathbb{R}_{>0}$ .
2  $m_1 \leftarrow \sum_{i=1}^m T_i$ ,  $m_0 \leftarrow m - m_1$ ;
3 Let  $(X_i^0, Y_i^0)$  be the  $i^{\text{th}}$  data point of type zero, and  $(X_i^1, Y_i^1)$  be the  $i^{\text{th}}$  data point of
   type one;
4 Create array  $Z$  of length  $\min\{m_0, m_1\}$ , with values  $Z_i = (\theta^\top X_i^0 - Y_i^0) - (\theta^\top X_i^1 - Y_i^1)$ ;
5 return  $\max\{\text{TTestUpperBound}(Z, \delta/2), \text{TTestUpperBound}(-Z, \delta/2)\}$ ;

```

Figure 4.9: TTestDiscrimUpperBound($\theta, D, \delta, \epsilon$) [1]

5. QNDLR(λ)

The QNDLR algorithm splits the dataset $D = \{(X_i, Y_i, T_i)\}_{i=1}^m$ into two parts: D_1 (20%) for training and D_2 (80%) for validation. It selects a candidate parameter vector θ_c by minimizing the TTestCandidateObjective on D_1 , which balances prediction error and fairness using the t -test bound and a weight λ . The value of λ determines the trade-off between accuracy and fairness: higher values place greater emphasis on the fairness component of the objective function, while lower values prioritize predictive performance. The fairness of θ_c is then assessed on D_2 via the TTestDiscrimUpperBound. If the upper bound meets the discrimination threshold ϵ , the candidate is returned; otherwise, no solution is found.

```

1 Assumptions: This quasi-Seldonian algorithm assumes that the sample
   discrimination statistic computed using all of the data is normally distributed, and
   also uses the minimum MSE estimator of variance within Student's  $t$ -test rather than
   the unbiased estimator.
2 Input: 1) Data set  $D = \{(X_i, Y_i, T_i)\}_{i=1}^m$ , 2) confidence level  $\delta \in (0, 1)$ , 3) maximum
           level of discrimination  $\epsilon \in \mathbb{R}_{>0}$ , 4) a hyperparameter  $\lambda \in \mathbb{R}_{\geq 0}$ .
3 Partition  $D$  into  $D_1$  (20% of data) and  $D_2$  (80% of data);
4  $\theta_c \in \arg\min_{\theta \in \Theta} \text{TTestCandidateObjective}(\theta, D_1, \delta, \epsilon, |D_2|, \lambda)$ ;
5 if  $\text{TTestDiscrimUpperBound}(\theta_c, D_2, \delta, \epsilon) \leq \epsilon$  then
6   return  $\theta_c$ ;
7 return NO SOLUTION FOUND;

```

Figure 4.10: Quasi-Non-Discriminatory Linear Regression (QNDLR if $\lambda = 0$ and QNDLR(λ) if $\lambda > 0$) [1]

Python Implementation of QNDLR: Technical Highlights

The implementation of QNDLR follows the same core structure as NDLR, with all aspects such as random partitioning, vectorized computations, and Nelder-Mead optimization implemented exactly as in NDLR.

The only difference is the **Feature Dimension Handling**: QNDLR dynamically detects

the feature dimension by inspecting the shape of the first data point in the training split, allowing support for both one-dimensional and multi-dimensional features. This determines the size of the initial parameter vector θ .

This modification was made to enable the use of QNDLR on datasets with multiple features.

In the next chapter, we will test the Seldonian algorithms using the Python implementations on both synthetic and real data.

Chapter 5

Results

In this section I present and interpret the results of applying the algorithms NDLR and QNDLR on both syntehtic and real datasets. The performance of NDLR and QNDLR is compared using a set of experiments. Finally I provide a comparative analysis between QNDLR and the SOTA algorithm, Grid Search Reduction

5.1 Applying NDLR and QNDLR to the Illustrative

Example

To verify the correctness of the Python implementation and ensure the NDLR and QNDLR functions perform as intended, I applied them to the illustrative example presented in the previous chapter.

5.1.1 Experimental Setup

From the synthetic dataset introduced earlier, I randomly sampled 500,000 points. I then applied the NDLR function using this dataset as input, along with the parameters $\delta = 0.05$, $\epsilon = 0.1$, and $b = 1.5$. The function returned an optimal parameter θ . I used θ to predict the target values and i implemented a function to compute the discrimination statistic $d(\theta)$ in order to assess the fairness of the resulting model

Similarly, I applied the QNDLR function with the same inputs and set $\lambda = 0$. This also returned an optimal θ used for prediction.

5.1.2 Results

This section presents the outcomes of applying the NDLR and QNDLR algorithms to the synthetic dataset described earlier. The results demonstrate how well these models achieved fairness while maintaining predictive accuracy.

NDLR Application – Results

Using the synthetic dataset D , the NDLR function returned an optimal parameter $\theta_c = 0.95488412$.

With this parameter, predicted target values $\hat{Y}$ were generated and the regression line was plotted. Figure 5.1 shows 5,000 randomly sampled points from D alongside the optimal regression line produced by NDLR.

I then calculated the discrimination statistic $d(\theta)$ to evaluate fairness. The result, $d_\theta = -0.09$, confirms that the fairness constraint $\epsilon = 0.1$ is satisfied, as NDLR guarantees. On the contrary, ordinary least squares linear regression resulted in a discrimination statistic of approximately 0.67.

Figure 5.2 visually contrasts the NDLR regression line (green dotted) with the least squares regression line (black solid). It is clear that NDLR mitigates the systematic bias present in the standard regression by adjusting overprediction and underprediction across sensitive groups.

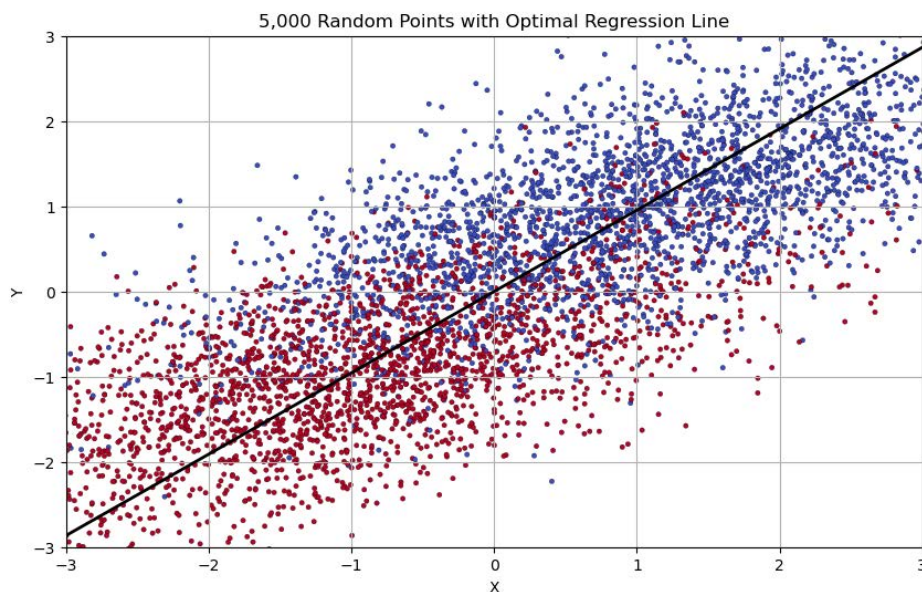


Figure 5.1: 5,000 randomly sampled points from D with the optimal regression line returned by NDLR.

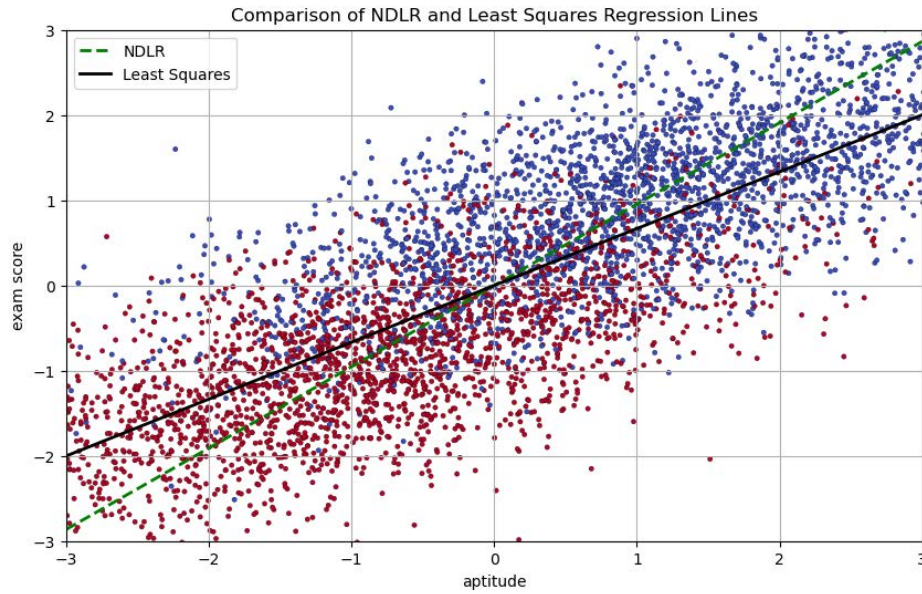


Figure 5.2: Comparison of NDLR (green dotted) and least squares (black solid) regression lines. Points are colored by sensitive group T .

QNDLR Application – Results

Similarly, I applied the QNDLR function with the same inputs and set $\lambda = 0$. This also returned an optimal $\theta_c = 0.95975586$ used for prediction. Predictions using θ_c yielded a discrimination statistic of $d_\theta = -0.08$.

Figure 5.3 shows the regression line from QNDLR, and Figure 5.4 compares regression lines from NDLR (green dashed), QNDLR (blue dashed), and least squares (black solid). The Seldonian algorithms produce nearly identical, overlapping regression lines.

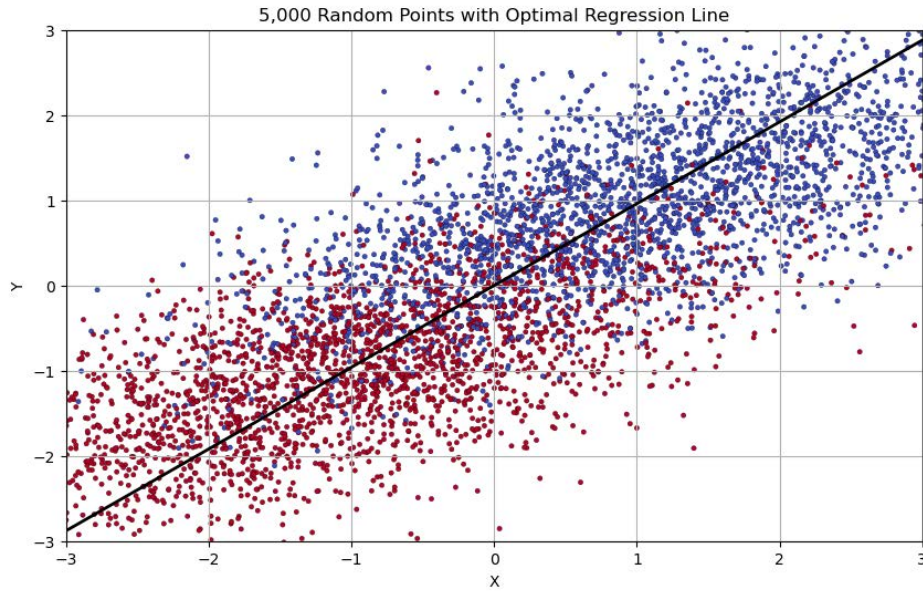


Figure 5.3: 5,000 randomly sampled points from D with the optimal regression line returned by QNDLR.

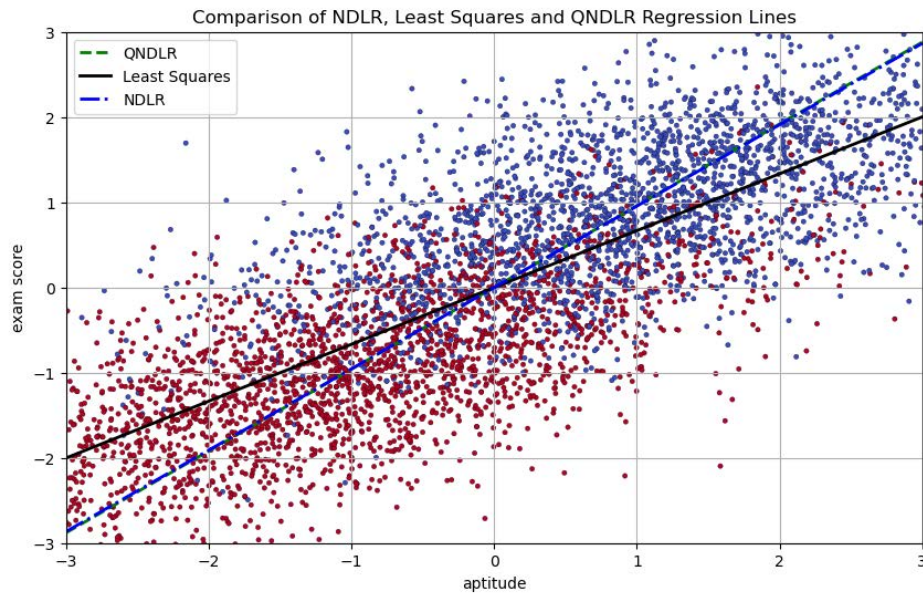


Figure 5.4: Comparison of regression lines from NDLR (green dashed), QNDLR (blue dashed), and least squares (black solid). Points are colored by sensitive group T .

5.1.3 NDLR & QNDLR Experiments

To further analyze the behavior of the Seldonian algorithms, I generated additional plots examining their prediction error and data efficiency. These experiments compared NDLR and

QNDLR to each other and to standard linear regression under training set sizes of 10^2 , 10^3 , 10^4 , 10^5 , and 10^6 .

Algorithms' Robustness vs. Training Size

I evaluated how often NDLR and QNDLR successfully return a valid solution across training set sizes of 10^2 , 10^3 , 10^4 , 10^5 , and 10^6 . For each size m , both algorithms were run 100 times on randomly sampled subsets of D , and the proportion of successful solutions was recorded.

Figure 5.5 illustrates that neither algorithm returns solutions with fewer than 10^3 training points. At $m = 10^4$, QNDLR produces valid solutions in roughly 45% of trials, and NDLR in about 41%. Increasing to $m = 10^5$ improves QNDLR's success rate to about 63%, while NDLR remains near 41%. At $m = 10^6$, QNDLR succeeds nearly always, while NDLR reaches around 50%. This indicates QNDLR's superior data efficiency.

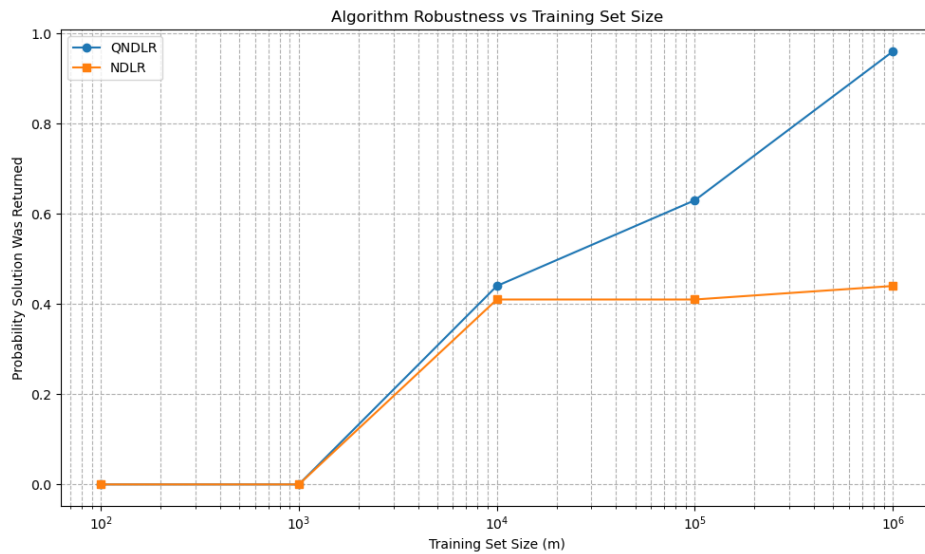


Figure 5.5: Probability that NDLR and QNDLR return a solution as a function of training set size m , averaged over 100 trials per m .

Probability of Over-Discrimination

I compared the frequency with which Linear Regression, NDLR, and QNDLR produced solutions whose discrimination statistic exceeded the fairness threshold $\epsilon = 0.1$ across different training sizes. For each m , multiple trials were run and the proportion of violations recorded.

As shown in Figure 5.6, Linear Regression consistently produces solutions exceeding the discrimination threshold across all m . Both NDLR and QNDLR maintain discrimination within bounds in all trials, confirming their fairness guarantees. QNDLR's guarantee depends on normality assumptions, allowing a small probability of violation in rare cases .

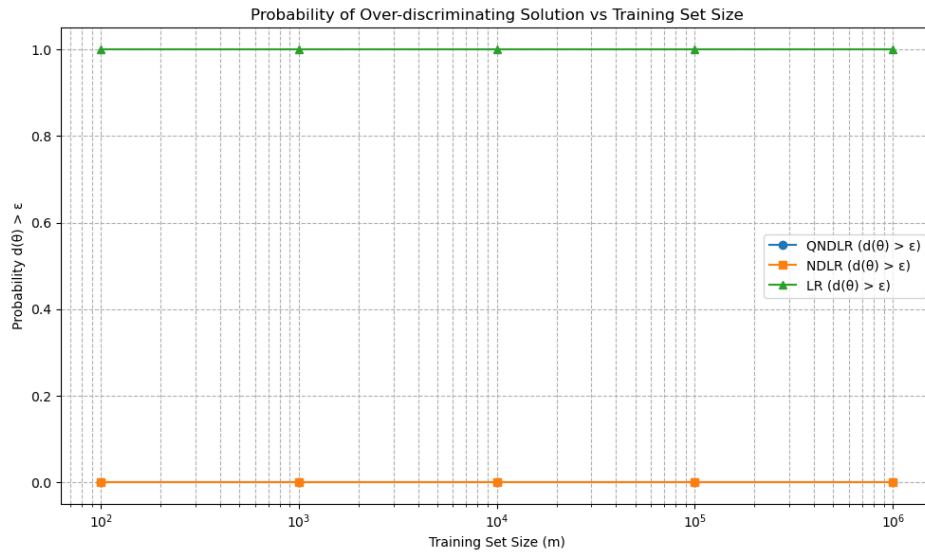


Figure 5.6: Probability that discrimination statistic exceeds $\epsilon = 0.1$ for Linear Regression, NDLR, and QNDLR across training sizes.

Mean Discrimination Statistic vs. Training Size

For each training size m , I computed the average discrimination statistic $d(\theta)$ across all successful runs of each algorithm, illustrating how fairness evolves with more training data.

Figure 5.7 shows that Linear Regression's discrimination statistic remains high and biased regardless of training size. NDLR and QNDLR significantly reduce discrimination, stabilizing near the fairness threshold as training size increases.

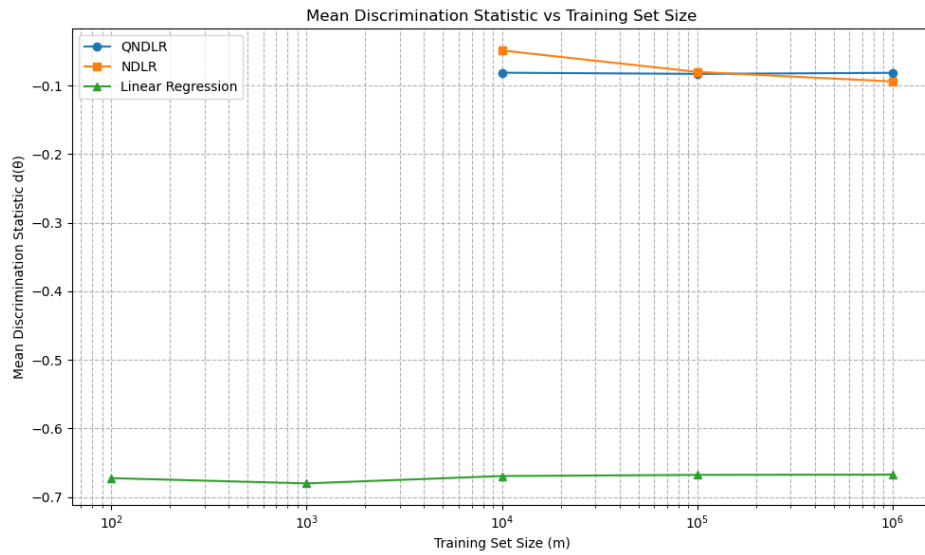


Figure 5.7: Mean discrimination statistic $d(\theta)$ for Linear Regression, NDLR, and QNDLR across training sizes.

Mean Squared Error (MSE) vs. Training Size

I evaluated the predictive accuracy of Linear Regression, NDLR, and QNDLR by measuring the MSE on a fixed test set as m varied. Results were averaged over successful runs for each algorithm and training size.

Figure 5.8 demonstrates that Linear Regression achieves consistently lower MSE, as it optimizes purely for accuracy without fairness constraints. NDLR and QNDLR return solutions only for larger training sizes, showing higher MSE but effective fairness enforcement. This highlights the trade-off between fairness and accuracy in these methods.

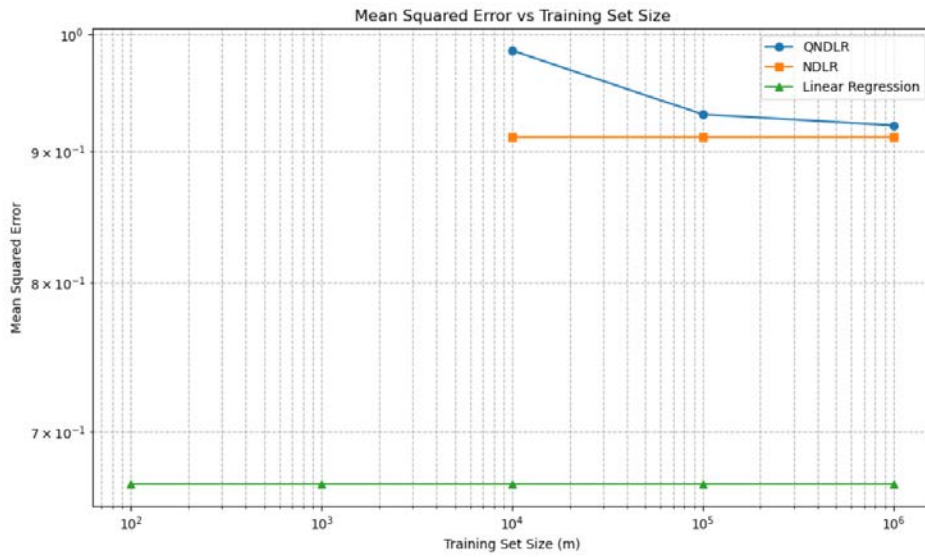


Figure 5.8: Mean Squared Error for Linear Regression, NDLR, and QNDLR across training sizes.

5.2 Seldonian Algorithms and Real Life Data

Having validated the Seldonian algorithms on synthetic datasets, where conditions can be carefully controlled, we now turn our attention to real-world data. Applying these algorithms to real data is crucial to demonstrate their practical effectiveness and robustness in authentic, complex scenarios. This section details the experimental setup used to evaluate the algorithms on the *UCI Law School GPA* [22], highlighting the challenges and considerations that arise when moving from simulated to real data. It further presents and analyzes the obtained results.

5.2.1 Experimental Setup

For the evaluation of the Seldonian algorithms, I utilize the *UCI Law School GPA* dataset obtained from the AIF360 repository. This dataset is widely used in fairness-aware machine learning research due to its inclusion of sensitive attributes and real-world performance outcomes.

Dataset Description

The dataset contains 22,342 records of law school students, including two primary predictive features: `lsat`, which corresponds to the Law School Admission Test score, and

ugpa, representing the undergraduate grade point average. The target variable of interest is *zfygpa*, the z-scored first-year law school GPA, which standardizes students' first-year academic performance to have zero mean and unit variance.

- **LSAT (Law School Admission Test Score):** The LSAT scores range approximately from 15 to 48, exhibiting a right-skewed distribution. Most students score in the mid to upper 30s, with a mode between 35 and 40 where the highest concentration of observations (over 7000) occurs. This distribution indicates that while there are fewer low scores below 25 and extremely high scores above 45, the majority of students cluster around the higher scoring range.
- **UGPA (Undergraduate GPA):** The UGPA values range from 2.0 to 4.0 on a typical GPA scale and show a slightly right-skewed distribution. The highest frequency of UGPA scores lies between 3.3 and 3.5, with a strong concentration in the 3.0 to 3.7 range. This suggests that most law school applicants have above-average undergraduate academic performance, with very few students having GPAs below 2.5.
- **ZFYGPA (Z-scored First Year GPA):** The target variable, ZFYGPA, has been standardized to have zero mean and unit variance, resulting in a symmetric, approximately normal (bell-shaped) distribution. The values range roughly from -4 to 3.5, with the majority of data points falling between -2 and +2. This standardized target facilitates the regression modeling task by ensuring the dependent variable has a well-behaved distribution.

In summary, the dataset features and target provide meaningful, interpretable information for predicting law school performance while considering the underlying distribution shapes and central tendencies. This careful characterization aids in understanding the data and informs the choice of appropriate modeling techniques.

Sensitive Attribute

The sensitive attribute considered in this experiment is *Race*. This attribute is the one used to identify and mitigate bias in the predictive models. Although the dataset also includes *Sex* as a potential sensitive attribute, preliminary analysis revealed no significant evidence of discrimination based on sex. This highlights the importance of carefully selecting sensitive

attributes relevant to fairness constraints, as different attributes may exhibit varying degrees of bias within the same dataset.

Methods Compared

Among the Seldonian algorithms, only the QNDLR (Quasi-Non-Discriminatory Linear Regression) algorithm is applied to the real-world dataset in this study. The original NDLR (Non-Discriminatory Linear Regression) algorithm relies on strict statistical guarantees based on Hoeffding's inequality, which require large sample sizes to produce meaningful results. In practical scenarios with limited data, such as the UCI Law School GPA dataset, these strict bounds often cause the NDLR algorithm to frequently return *No Solution Found* (NSF).

Consequently, QNDLR, which employs relaxed statistical assumptions and quasi-non-discriminatory constraints, is more suitable for datasets of this size and is therefore the focus of this experimental evaluation. Specifically, two variants of QNDLR are tested: one with $\lambda = 0$ and one with $\lambda = 1$. For both variations of QNDLR, the fairness tolerance was set to $\epsilon = 0.1$ and the confidence level to $\delta = 0.05$. This means the algorithm ensures, with at least 95% confidence, that the upper bound on the discrimination metric does not exceed 0.1. In addition to QNDLR, we also compare the performance of the standard Ordinary Least Squares (OLS) Linear Regression.

Goal

The primary goal is to predict first-year law school GPA while minimizing racial bias in the predictions.

This experimental setup provides a realistic and challenging environment to test the effectiveness of Seldonian algorithms in balancing predictive performance with fairness requirements.

5.2.2 Results

The performance of the compared models on the UCI Law School dataset is summarized below:

- **Ordinary Least Squares (OLS) Linear Regression**

- Discrimination statistic $d(\theta)$: 0.5701
- Mean Squared Error (MSE): 0.8552
- **QNDLR ($\lambda = 0$)**
 - Optimized parameters $\theta_{\text{opt}} = [0.1207, 0.3225]$
 - Discrimination statistic $d(\theta)$: -0.0290
 - Mean Squared Error (MSE): 30
- **QNDLR ($\lambda = 1$)**
 - Optimized parameters $\theta_{\text{opt}} = [0.1225, 0.0540]$
 - Discrimination statistic $d(\theta)$: 0.0592
 - Mean Squared Error (MSE): 22

5.2.3 Analysis of Results

The discrimination statistic $d(\theta)$ clearly illustrates the key difference between ordinary linear regression and the QNDLR models. The linear regression model exhibits substantial discrimination against non-white students, with a high $d(\theta)$ value of approximately 0.57, indicating biased predictions. In contrast, both QNDLR variants maintain discrimination well below the fairness threshold of 0.1, consistent with their theoretical guarantees, regardless of the λ value used.

It is important to note that the Mean Squared Error (MSE) values returned by the QNDLR algorithm (for both $\lambda = 0$ and $\lambda = 1$) cannot be directly compared to the MSE obtained from ordinary linear regression. This discrepancy arises because QNDLR does not estimate an intercept term in the model. While this omission did not affect previous experiments on synthetic data—where both features and targets were mean-centered (mean = 0)—it becomes significant here because only the target variable `zfygpa` is standardized (z-scored) with mean zero, whereas the features (`lsat`, `ugpa`) are not. Ordinary Least Squares regression includes an intercept to account for this shift, which reduces prediction error.

Therefore, direct comparisons between OLS and QNDLR MSE values are not meaningful in this setting. However, comparisons between the two QNDLR variants remain valid since both omit the intercept and operate under the same assumptions. Notably, the soft-constrained

variant with $\lambda = 1$ achieves a lower MSE (22) compared to the hard-constrained variant $\lambda = 0$ (MSE = 30), demonstrating that incorporating the fairness penalty within the objective allows the model to find a better trade-off between predictive accuracy and fairness.

5.3 QNDLR vs. Grid Search Reduction

The evaluation of novel bias reducing algorithms like QNDLR relative to robust State-Of-The-Art techniques is a critical step in the process of deploying new methods for fair regression models. The comparison with highly used and adopted approaches such as Grid Search Reduction [14], ensures new techniques succeed, compete, uphold theoretical guarantees and retain practical usefulness.

5.3.1 Experimental Setup

To compare our developed QNDLR algorithm against current state-of-the-art methods, I employed the Grid Search Reduction (GSR) method provided within the AI Fairness 360 (AIF360) library. Code and pipeline also existed within a public example notebook¹, released by AIF360, that showcases the application of GSR for fair regression tasks. The GSR method, initially described by Agarwal et al. [14], is detailed further in the Background section.

Dataset and Preprocessing

The Law School GPA dataset, previously described, was applied across all experiments. It was loaded in an scikit-learn-compatible format via AIF360's data retrieval tools. Both training and test sets include the features `LSAT` and `uGPA`, `race` as the protected feature and `zfygpa` as the target.

Continuous features were normalized using `MinMaxScaler` to scale values into the $[0, 1]$ range. Importantly, the dataset indices retained the protected attribute values (`race`). This enables each data point's group identification without the inclusion of the protected attribute in the feature set. By externally recording group membership the algorithm is allowed to enforce fairness constraints, such as bounded group loss, by monitoring performance across groups, ensuring that fairness is properly measured and adjusted during training.

¹AIF360 GSR Regression Notebook

Baseline Model and Metrics

A linear regression model was trained on the normalized training data. Of course the protected attribute was excluded to prevent direct discrimination. The model was wrapped in a `TransformedTargetRegressor` for consistent scaling of the target variable. The performance of the model was estimated on the test set using mean absolute error (MAE), and fairness was evaluated in terms of the protection group difference in MAE. The difference in MAE was calculated by using AIF360's `difference` metric.

Grid Search Reduction Setup

The Grid Search Reduction algorithm was instantiated with a linear regression base estimator, wrapped in a `TransformedTargetRegressor` with scaling.

Along with the absolute loss function, `BoundedGroupLoss` was employed as a fairness constraint. The difference in MAE between protected groups was used to assess fairness.

The algorithm was given the target variable's range (`min_val` and `max_val`) to suitably bound the loss throughout the search.

The grid size was set to 10 for searching over candidate models. The protected feature `race` was excluded from input features at fitting time.

Reproducibility

For the sake of reproducibility, a random seed was set before model training. The model was trained on the training dataset and tested on the test dataset. The resultant MAE and fairness metric (difference of MAE between groups) were computed and compared to the baseline outcome.

Extension for QNDLR Comparison

Some significant changes to the original notebook were necessary in order to effectively compare QNDLR's performance to the GSR approach as implemented in `aif360` library.

QNDLR Dataset Preparation

Since QNDLR requires input in the form of triplets $[X_i, Y_i, T_i]$, where T_i denotes group membership, I constructed a dataset `D_formatted` that combines the scaled features, scaled

target values, and protected group labels. Moreover, I ensured that the scaling of the training target values was aligned with the internal optimization performed by QNDLR

QNDLR Alterations

Two main alterations were required to enable a fair and consistent comparison between QNDLR and the state-of-the-art GSR method.

First, I modified the QNDLR implementation in a way that it minimizes the mean absolute error (MAE) instead of the mean squared error (MSE).

Second, I introduced a new variant of the QNDLR algorithm that uses the *maximum MAE difference between groups* as the fairness constraint instead of the original discrimination statistic, which was defined as:

$$d(\theta) := \mathbb{E}[u(y(X, \theta) - Y) \mid T = 0] - \mathbb{E}[u(y(X, \theta) - Y) \mid T = 1]$$

This variant is referred to throughout as *QNDLR_MAE_DIFF*.

QNDLR Execution

I incorporated calls to both the QNDLR and *QNDLR_MAE_DIFF* functions and executed training with the following parameters:

- For QNDLR: constraint $\epsilon = 0.1$, confidence level $\delta = 0.05$, and bound $b = 1.5$
- For *QNDLR_MAE_DIFF*: constraint $\epsilon = 0.07$, confidence level $\delta = 0.05$, and bound $b = 1.5$

In both cases, a retry loop was employed to handle possible "No Solution Found" (NSF) returns, ensuring a valid model was eventually obtained.

The choice of $\epsilon = 0.07$ for *QNDLR_MAE_DIFF* was motivated by the observation that the MAE difference from the GSR model was approximately 0.06; setting ϵ slightly above this value allows us to examine if QNDLR can match or outperform GSR under similar fairness constraints.

Here I must note that MAE difference evaluations are performed using *unscaled* target predictions and test values. Therefore, it was necessary to scale the fairness threshold ϵ passed to *QNDLR_MAE_DIFF*. This was done by dividing the unscaled threshold by the range of

the original target variable:

$$\text{scaled_epsilon} = \frac{\text{unscaled_epsilon}}{y_{\text{train_max}} - y_{\text{train_min}}}$$

This ensures that the fairness constraint is expressed in the same scale as the normalized targets used in QNDLR's optimization, allowing for correct constraint enforcement.

Prediction and Evaluation

I used the optimal parameter θ^* obtained from QNDLR to make predictions on the scaled test set. I then inverse-transformed the outputs to the original target scale. Next, I computed MAE, group-wise MAE, and fairness metrics, including the discrimination statistic and MAE difference, to evaluate performance.

This setup allowed a fair and direct comparison of our QNDLR algorithm with the GSR method in terms of both predictive performance and group fairness on a real-world regression task.

Comparison

The final comparison included the following models:

- The baseline linear regression model
- The GSR model with bounded group loss
- The QNDLR model with the original fairness constraint
- The *QNDLR_MAE_DIFF* model with group MAE difference constraint

5.3.2 QNDLR vs. GridSearchReduction - Results

In this section, I present the results of the QNDLR algorithm, opposed to the GSR method. The comparison considers both performance and fairness.

Table 5.1 are summarizes the metrics used to assess performance and fairness for the four regression methods evaluated: Ordinary Least Squares (OLS), Grid Search Reduction (GSR), QNDLR, and the variant *QNDLR_MAE_DIFF*.

Method	Statistical Guarantees	MAE	MAE Difference
OLS	None	0.74	0.2
GSR	None	0.76	0.06
QNDLR	$\epsilon = 0.1, \delta = 0.05$	0.76	0.09
<i>QNDLR_MAE_DIFF</i>	$\epsilon = 0.07, \delta = 0.05$	0.8	0.01

Table 5.1: Performance and fairness comparison across methods.

The basic OLS model achieves the lowest mean absolute error (MAE) equal to 0.74. On the contrary it manages the highest group MAE difference at 0.2, number that indicates significant fairness disparities between the two groups.

The GSR method which relies on experimental validation rather than on strict statistical guarantees, returns a MAE of 0.76, slightly higher compared to what OLS achieves. Although, notably reduces the fairness gap to 0.06.

When it comes to the QNDLR model, recall that it was trained using a different discrimination statistic than the one evaluated. Despite this fact, it maintains comparable prediction accuracy (MAE=0.76) and on top of that improves fairness by reducing the MAE difference to 0.09.

Finally, the *QNDLR_MAE_DIFF* variant achieves the best fairness outcome, reducing the MAE difference to 0.01, at the cost of a slight increase in overall error (MAE = 0.8).

These results highlight the fundamental trade-off between accuracy and fairness. Nevertheless, methods that incorporate fairness constraints are capable of reducing significantly disparities with only minor sacrifices in error metrics. The *QNDLR_MAE_DIFF* model, in particular, shows strong potential for applications where fairness is crucial, successfully managing bias mitigation while maintaining acceptable performance.

Overall, QNDLR demonstrates encouraging results, comparable to GSR method. Along with competitive accuracy and fairness, QNDLR provides strong statistical guarantees and offers users the freedom to adjust the discrimination metric, fairness constraints and confidence levels. This versatility makes QNDLR a strong alternative to GSR in fair regression tasks.

Chapter 6

Conclusion

The research done for the thesis is summed up in this chapter. The key conclusions are examined, the contributions are emphasized, and the results' outcomes are mentioned. Furthermore, possible paths for further investigation are presented based on the study's findings

6.1 Summary and Conclusions

This thesis examined the application of the Seldonian framework, specifically employing the NDLR and QNDLR algorithms, to address fairness issues in regression problems. The Seldonian framework provides a principled approach to constructing safe and robust machine learning models through enforcing fairness constraints with robust statistical guarantees. The framework is also very flexible and can be readily generalized to other fairness measures, and the produced algorithms are competitive in both accuracy and fairness compared to best-in-class benchmarks.

The empirical evaluation revealed that both NDLR and QNDLR are effective at minimizing discrimination in predictions without sacrificing rational predictive accuracy. On synthetic data, the algorithms uniformly achieved pre-specified fairness constraints, significantly outperforming traditional least squares regression in fairness metrics. Both qualitative and quantitative comparisons of their ability to prevent systematic bias over sensitive subgroups confirmed their capability.

QNDLR was more stable and effective with limited data than NDLR, requiring fewer training samples to almost always provide feasible solutions. The application of QNDLR to real-world data further validated these findings, with both hard- and soft-constrained variants

maintaining discrimination well below the fairness threshold. The soft-constrained variant even provided improved fairness versus accuracy balance.

Despite the strengths, there were many practical limitations still. NDLR requires an enormous volume of training data to perform optimally, limiting its application in the majority of real-world situations. Additionally, Seldonian algorithms both omit an intercept term, thus are incompatible with datasets where features have not been standardized or z-scored accordingly. This deficiency can result in systematic prediction errors when applied to unscaled real-world data. Further, when applied to real datasets, QNDLR often would not provide a solution unless wrapped in an iterative search (e.g., a for-loop) increasing computational complexity and running time.

Comparison with the Grid Search Reduction method showed that QNDLR not only rivals it in prediction accuracy and fairness but also provides strong statistical guarantees and more flexibility in fairness definitions and confidence levels. This points to the Seldonian framework as a valuable and versatile framework for fair regression modeling.

In summary, the Seldonian approach offers a safe and trustworthy technique for the integration of fairness in machine learning, especially in regression problems. While the current algorithms are effective, it is necessary to tune and extend the methodologies further to improve their efficiency and compatibility with real-world data.

The principal contribution of this thesis is that the Seldonian framework is a very valuable addition to the fairness in regression literature. It adequately fills an essential gap through the provision of rigorous statistical guarantees of fairness as well as competitive predictive accuracy, addressing issues which cannot effectively be met by other approaches. This renders the framework an impactful and practical mechanism for effecting trustworthy and fair regression modeling. This thesis examined the application of the Seldonian framework, specifically employing the NDLR and QNDLR algorithms, to address fairness issues in regression problems. The Seldonian framework provides a principled approach to constructing safe and robust machine learning models through enforcing fairness constraints with robust statistical guarantees. The framework is also very flexible and can be readily generalized to other fairness measures, and the produced algorithms are competitive in both accuracy and fairness compared to best-in-class benchmarks.

6.2 Future Extensions

While the present application of the Seldonian framework shows encouraging outcomes in imposing fairness with statistical guarantees, there are a number of directions for future work to expand and strengthen the framework for more general use and better performance.

One of the most obvious additions is the inclusion of intercept terms to the regression models. The availability of intercepts can increase flexibility and accuracy of models, especially in applied problems when data is unscaled. Such an improvement might lead to better predictive accuracy with retained fairness constraints.

Moreover, there is significant potential in extending the Seldonian method to nonlinear models. The majority of real-world relationships are nonlinear, and allowing nonlinear regression or classification models such as kernel methods, decision trees, or neural networks would make the method much more widely applicable. It would render the comparison between seldonian algorithms and State of the Art more effective too. This generalization would entail developing novel algorithmic techniques for maintaining fairness guarantees on more complex model classes.

Together, these future changes will add flexibility, stability, and usability to the Seldonian framework. They may pave the way for more trustworthy, unbiased and fair machine learning models in myriad real-world applications.

Bibliography

- [1] Philip S. Thomas, Bruno Castro da Silva, Andrew G. Barto, Stephen Giguère, Yuriy Brun, and Emma Brunskill. Preventing undesirable behavior of intelligent machines. *Science*, 366(6468):999–1004, 2019.
- [2] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning: Limitations and Opportunities*. MIT Press, 2023. Available online at <https://fairmlbook.org>.
- [3] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 214–226, 2012.
- [4] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268. ACM, 2015.
- [5] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35, 2021.
- [6] Indrè Žliobaitė. A survey on measuring indirect discrimination in machine learning. *arXiv preprint arXiv:1511.00148*, 2015.
- [7] Alekh Agarwal, Miroslav Dudík, and Zhiwei Steven Wu. Fair regression: Quantitative definitions and reduction-based algorithms. In *International Conference on Machine Learning*, pages 120–129. PMLR, 2019.

- [8] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.
- [9] W. S. Gosset. The probable error of a mean. *Biometrika*, 6(1):1–25, 1908. Published under the pseudonym "Student".
- [10] Xuezhou Hu, Xueru Wu, and Virginia Smith. Fair federated learning via bounded group loss. In *International Conference on Learning Representations (ICLR)*, 2022.
- [11] Rachel KE Bellamy, Kuntal Dey, Michael Hind, Samuel C Hoffman, Stephanie Houde, Kalpesh Jain, Peter J John, Dimitris Kannan, Prasanna K Narayanan, Danish Ramamurthy, and John S Richards. Ai fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 2018.
- [12] IBM Research. Ai fairness 360: Open source toolkit to detect and mitigate bias in machine learning models. <https://www.ibm.com/blog/ai-fairness-360-open-source-toolkit/>, 2018.
- [13] Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. Fairlearn: A toolkit for assessing and improving fairness in AI. Technical Report MSR-TR-2020-32, Microsoft, May 2020.
- [14] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna Wallach. A reductions approach to fair classification. In *International Conference on Machine Learning*, pages 60–69. PMLR, 2018.
- [15] Fairlearn Developers. *fairlearn.reductions.GridSearch*, 2023. Fairlearn v0.5.0 documentation.
- [16] Furkan Gursoy and Ioannis A. Kakadiaris. Error parity fairness: Testing for group fairness in regression tasks. *arXiv preprint arXiv:2208.08279v1*, Aug 2022. <https://arxiv.org/abs/2208.08279v1>.
- [17] Dana Pessach and Erez Shmueli. A review on fairness in machine learning. *ACM Computing Surveys (CSUR)*, 55(3):1–44, 2023.

- [18] Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, pages 3315–3323, 2016.
- [19] Neda Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *arXiv preprint arXiv:1908.09635*, 2019.
- [20] Faisal Kamiran, Amina Karim, Xin Zhang, and Xia Zhang. A comparative study of fairness in machine learning. *arXiv preprint arXiv:2003.11116*, 2020.
- [21] Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.
- [22] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Shivani Suresh, Venkatesh Narayanan, Dinesh Ramamurthy, Prasanna Sattigeri, Katherine Chou, Nina Diakonikolas, Francesca Ferrario, and Himabindu Lakkaraju. aif360.datasets.lawschoolgpadatastet. <https://aif360.readthedocs.io/en/latest/modules/generated/aif360.datasets.LawSchoolGPADataset.html>, 2018–2024.

APPENDICES

Appendix A

Source Code Documentation

This appendix provides an overview of the source code developed during this thesis. It includes a detailed list of all scripts, notebooks, and modules used to implement the various stages of the research work. For each file, a concise description of its purpose and role within the project is given. Additionally, this appendix serves as a guide to help readers navigate the codebase, understand the interactions between different components, and identify the specific files corresponding to each experiment or section of the thesis. The aim is to facilitate reproducibility and enhance transparency regarding the implementation details underlying the research.

A.1 Code Navigation

Below is a detailed list of the main scripts and notebooks developed and used throughout this thesis. Each entry includes the filename along with a brief description of its functionality and role within the project.

- `synthetic_data_creation.ipynb`
Generates the synthetic dataset `all_synthetic_data2.csv` based on the illustrative example. This dataset is included in the repository.
- `OLS_training.ipynb`
Applies OLS to compute regression coefficients and discrimination statistics. Produces `discrimination_statistics_with_coefficients.csv`.
- `soft_constraints_exp.ipynb`

Experiments with soft fairness constraints. Requires TensorFlow.

- `ndlr.py`

Python implementation of the NDLR algorithm.

- `ndlr_application.ipynb`

Applies NDLR to synthetic data. Depends on `ndlr.py`.

- `QNDLR.py`

Implementation of QNDLR for one-dimensional features.

- `QNDLR_application.ipynb`

Applies QNDLR on synthetic data and includes experiments with NDLR and QNDLR.

Requires both `QNDLR.py` and `ndlr.py`.

- `QNDLR_MSE.py`

QNDLR implementation for two-dimensional features using Mean Squared Error (MSE).

- `qndlr_real_data.ipynb`

Applies QNDLR to real-world datasets, relying on `QNDLR_MSE.py`.

- `QNDLR_MAE.py`

QNDLR variant using Mean Absolute Error (MAE) loss.

- `QNDLR_MAE_DIFF.py`

QNDLR variant using MAE and the `MAE_diff` discrimination metric.

- `QNDLR_VS_GSR.ipynb`

Compares QNDLR methods to Group Statistical Relaxation (GSR). Requires `aif360` and calls both `QNDLR_MAE.py` and `QNDLR_MAE_DIFF.py`.

- `demo_grid_search_reduction_regression_sklearn.ipynb`

Unmodified demo notebook from AIF360's GitHub repository. Included for reference.

Do not alter.

A.2 Code-to-Thesis Mapping

Thesis Section / Experiment	Corresponding Code File(s)
Synthetic Data Generation 3.1.2	<code>synthetic_data_creation.ipynb</code>
OLS - Illustrative example 3.1.3	<code>OLS_training.ipynb</code>
Soft Constraints Experiment 3.2.4	<code>soft_constraints_exp.ipynb</code>
NDLR Algorithm 4.3.1	<code>ndlr.py</code>
QNDLR for 1D Features 4.3.2	<code>QNDLR.py</code>
NDLR, QNDLR on synthetic data 5.1	<code>QNDLR_application.ipynb</code> , <code>ndlr_application.ipynb</code>
QNDLR for 2D Features 4.3.2	<code>QNDLR_MSE.py</code>
QNDLR with real data 5.2.1	<code>qndlr_real_data.ipynb</code>
QNDLR Variants with MAE Loss 5.3.1	<code>QNDLR_MAE.py</code> , <code>QNDLR_MAE_DIFF.py</code>
QNDLR vs GSR 5.3.1	<code>QNDLR_VS_GSR.ipynb</code>

Table A.1: Mapping of thesis experiments and code files

A.3 Code Access

The complete source code for all experiments and algorithms discussed in this thesis is publicly available on GitHub at the following link: <https://github.com/elen011/Diploma-thesis-Fairness-in-ML.git>

Note on reproducibility: Some methods such as NDLR and QNDLR are stochastic and their outputs vary slightly across runs. To reproduce the exact results presented in this thesis, please use the pre-generated output files included in the repository rather than re-running the corresponding notebooks or scripts.

A.3.1 Dependencies:

Python version 3.6 or higher is required, along with the following packages: `numpy`, `pandas`, `matplotlib`, `scikit-learn`, `tensorflow`, and `aif360`.