Les membres du jury qui ont évalué ce mémoire ont demandé des corrections mineures.

HEC MONTRÉAL

Decision-Focused Learning for Multi-Stakeholder Recommender Systems

par

Audrey-Anne Guindon

HEC Montréal

Directeur/Directrice de recherche

Laurent Charlin

Sciences de la gestion
Data Science and Business Analytics

Mémoire présenté en vue de l'obtention
du grade de maîtrise ès sciences en gestion
(M. Sc.)

12, 2023

Résumé

De nombreuses entreprises en ligne opèrent en tant que plateformes multi-faces, réunissant diverses parties prenantes afin de réduire les coûts de recherche et de transaction. Les recommandations personnalisées jouent un rôle essentiel dans l'amélioration de l'engagement des utilisateurs sur ces plateformes, car le succès économique de ces entreprises repose sur leur capacité à attirer des annonceurs, des fournisseurs de contenu et d'autres parties prenantes qui bénéficient d'une attention accrue de la part des utilisateurs. La recommandation dans ce domaine implique de trouver un équilibre entre la pertinence pour l'utilisateur et la promotion équitable des fournisseurs. Sur la base de travaux antérieurs, nous explorons l'application de l'apprentissage axé-sur-la-décision, un paradigme émergent qui intègre la prédiction et l'optimisation contrainte dans un seul pipeline, au domaine de la recommandation multipartite. Nous proposons une approche qui fusionne le filtrage collaboratif neuronal, une technique courante d'apprentissage approfondi pour la recommandation personnalisée, avec des méthodes d'apprentissage équitable de classement en traitement interne. Nous étudions l'influence de cette approche sur l'équilibre entre l'équité et la pertinence par rapport aux approches en deux étapes. En outre, nous évaluons l'efficacité de ces méthodes dans des scénarios impliquant plusieurs parties prenantes, où notre objectif est de maximiser l'utilité pour les utilisateurs tout en représentant tous les fournisseurs sur la plateforme. Nos résultats confirment que ces techniques de classement équitable maintiennent des performances élevées lorsqu'elles sont appliquées à des modèles à facteurs latents, et que l'apprentissage axé-sur-la-décision permet un meilleur contrôle du compromis entre la pertinence et l'équité.

Mots clés : Apprentissage axé-sur-la-décision ; Recommandation multipartite ; Recommandation équitable ; Classement ; Apprentissage automatique ; Apprentissage approfondi ; Optimisation

Méthodes de recherche : Recherche quantitative

Abstract

Many online businesses today operate as multi-sided platforms, which create value by bringing together buyers, sellers, and other partners to reduce search and transaction costs. Personalised recommendation plays a crucial role in enhancing user engagement and retention on these platforms, as the economic success of these multi-sided businesses relies on attracting advertisers, content providers, and other stakeholders who benefit from increased user attention. Effectively managing recommendations in a multi-sided platform is complex, requiring a balance between user relevance and fair exposure for providers. Building on previous work, we explore the application of decision-focused learning, an emerging paradigm in machine learning that integrates prediction and constrained optimization, to the domain of fair multi-stakeholder recommendation. We propose a framework that merges neural collaborative filtering, a prevalent deep learning technique for personalised recommendation, with in-processing fair learning to rank methods in an end-to-end pipeline. We study the effect of end-to-end learning in balancing fairness and accuracy in comparison to two-stage approaches. Additionally, we evaluate whether these methods are effective in multi-stakeholder scenarios, where our aim is to maximise the utility for a user, while simultaneously representing all the providers on the platform. Experimental results confirm that end-to-end fair learning to rank techniques maintain a high performance when applied to latent factor models, and that decision-focused learning offers better control over the tradeoffs between accuracy and fairness when compared to other end-to-end fair ranking methods.

Keywords : Decision-Focused Learning; Multi-Stakeholder Recommender Systems; Fair Recommendation; Learning to Rank; Machine Learning; Deep Learning; Constrained Optimization

Research methods : Quantitative Research

Table of Contents

List of Figures

List of Tables

Acknowledgment

It takes a long time to finish a Master's thesis when working full time or during a worldwide pandemic (4 years to be exact). Thank you to my parents for supporting me and gently reminding me that I needed to finish this work and move on to greater things. I am also immensely grateful to my beautiful fiance Aethelind for her love and support.

I would also like to thank my supervisor Laurent Charlin for his advice and guidance throughout this journey.

Chapter 1 Introduction

1.1 Study Context

The rapid expansion of online platforms, including e-commerce websites, online marketplaces, streaming services, and social media platforms, has presented users with a challenge when it comes to finding content that aligns with their preferences. To address this issue, recommender systems have emerged as a solution to alleviate information overload and enhance user experience by providing tailored recommendations.

Recommender systems have become an integral part of online platforms. However, traditional recommender systems often prioritise optimising user satisfaction while neglecting the interests and constraints of other stakeholders involved. In recent years, there has been an increasing recognition of the importance of considering the perspectives and needs of multiple stakeholders when making recommendations [2, 17, 99]. Many online services today operate as multi-sided platforms, which create value by bringing together buyers, sellers, and other partners to reduce search and transaction costs. Well-known examples of multi-sided platforms include Amazon, Etsy, eBay, Ticketmaster, Uber, LinkedIn, and Facebook.

Effectively managing recommendations for a multi-sided platform is a complex task that requires recommending items that offer high utility to consumers while ensuring that providers receive sufficient recommendations to perceive value in participating within the platform's network [76]. By considering the perspectives and constraints of both users and providers, recommender systems can enhance user satisfaction, promote provider engagement, and contribute to maintaining a thriving ecosystem within multi-sided platforms [74].

1.2 Problem & Research Question

We are tackling the problem of multi-stakeholder recommendation. The key task in multi-stakeholder recommendation is to provide recommendations that maximise utility for users while also ensuring representation for all the providers on the platform. However, achieving this

balance is not straightforward due to inherent biases in the data and the complex nature of user preferences.

The challenge arises from the fact that traditional recommendation algorithms often prioritise accuracy metrics, such as maximising user satisfaction or click-through rates, without explicitly considering fairness. Standard recommender systems suffer from popularity bias, favouring already popular items and perpetuating "rich get richer" dynamics [2]. As a result, certain providers may be consistently overrepresented, while others are overlooked, leading to provider attrition and a subsequent reduction in the diversity of items on the platform [54]. Conversely, a naive approach like recommending the least exposed products to increase provider exposure often leads to irrelevant recommendations, resulting in lower user utility [63], which undermines the purpose of the recommender system. Dealing with these algorithmic biases pose significant challenges, leading to the emergence of fair machine learning techniques.

A compelling strategy for addressing algorithmic bias involves incorporating constraints that mitigate bias directly into the recommendation algorithm. In-processing methods for fair machine learning are effective techniques that promote fairness within the learning process itself [102]. For fair recommendation, in-processing methods have been found to achieve better tradeoffs between accuracy and fairness compared to post-processing, because finding this balance is at the core of their learning objective [24, 95]. Notably, in-processing methods are capable of handling different types of underlying biases without requiring prior knowledge of the specific type present.

Regularisation-based approaches are a prominent group of in-processing methods for improving fairness and reducing bias in recommendations. Regularisation entails adding a term to the loss function of the learning model. These regularisation terms express measures of unfairness that the model must minimise in addition to minimising the original loss function [136]. A second multiplier term λ is used to control the strength of the regularisation term on the training objective. One of the challenges with regularisation-based approaches is setting the parameter λ. The parameter λ must be treated as a hyperparameter, increasing the computational effort required to find desirable solutions. Additionally, when a tradeoff between fairness and utility is

desired, it cannot be controlled by specifying an allowable magnitude for fairness violations because of the lack of a reliable relationship between the hyperparameter λ and the resulting violation [49].

Constraint-based approaches, whether employed as a post-processing step or integrated during training, form another prominent group of methods for fair recommendation [100]. Numerous studies have proposed constrained optimization as a means to achieve an optimal equilibrium between relevance and fairness.

Decision-focused learning is an emerging paradigm in machine learning that integrates prediction and constrained optimization into an end-to-end system [84]. It has proven to be competitive in solving various real-world problems that involve constrained optimization on predictions. Several problems that have benefited from decision-focused learning, such as ranking [11], top-k selection [57], diverse bipartite matching [84], and the knapsack problem [101], are relevant to the domain of recommendation [56]. Given the prevalence of constraint-based approaches in fair recommendation research, it is reasonable to explore the application of decision-focused learning in this context. However, there has been little research investigating their application to recommender systems, let alone to the task of fair recommendation.

The most relevant work we could find was conducted by Kotary et al. [49], which introduces *Smart "Predict, then Optimise" for Fair Ranking (SPOFR)*, a decision-focused learning approach that integrates optimization and fairness-constrained learning to rank. The proposed approach circumvents the challenges associated with other end-to-end learning approaches that penalise constraint violations during learning by utilising λ, and guarantees that both relevance and fairness objectives are met on each predicted ranking policy. To our knowledge, it is the only existing work exploring the application of decision-focused learning paradigms to the task of fair learning to rank.

However, their work focuses on an un-personalised information retrieval setting, which is distinct from personalised recommendation. Whereas information retrieval systems rely on explicit user queries or search terms to retrieve relevant documents, personalised

recommendation is more passive and requires learning user preferences and behaviour, often from latent factors that are not directly observable and must be inferred indirectly.

Latent factor models, like matrix factorization [48], are widely used due to their ability to produce good low-dimensional latent feature representations of users and items. Neural collaborative filtering [41] belongs to a category of embedding factorization models that incorporate deep neural networks. One key advantage of neural collaborative filtering is its compatibility with state-of-the-art in-processing fair learning to rank techniques.

Therefore, the objective of this thesis is twofold. First, we aim to determine whether decision-focused learning can be successfully applied to the domain of fair recommendation in order to improve the balance between accuracy and fairness. Second, we seek to establish the effectiveness of balancing accuracy and fairness in multi-stakeholder scenarios, where our aim is to maximise the utility for a user, while simultaneously fairly representing all the providers on the platform.

1.3 Potential Contributions to the Field

To the best of our knowledge, this work represents the first endeavour to merge neural collaborative filtering, a prevalent deep learning technique for personalised recommendation, with in-processing fair learning to rank methods. Furthermore, it is the first study to specifically investigate the application of decision-focused learning to the personalised recommendation task. Additionally, it is one of a limited number of works that explore in-processing fair machine learning methods for multi-stakeholder recommendation.

1.4 Dissertation Flow

The remainder of this thesis is organised as follows: Chapter 2 provides an overview of the preliminaries and background for this thesis, covers fundamental concepts and techniques for fair recommendation, and highlights the importance of considering multiple stakeholders in the design of recommender systems. Chapter 3 presents a comprehensive review of the state-of-the-art algorithms employed in fair learning to rank and proposes an approach to

combining these algorithms with latent factor models using deep learning. Chapter 4 focuses on our methodology and outlines our experimental framework. Chapter 5 reports and discusses the results of our experiments. Finally, Chapter 6 concludes the thesis by summarising the findings, and highlights possible future works.

Chapter 2 Preliminaries and Background

In this chapter, we provide an overview of the preliminary concepts and background knowledge that form the foundation of this thesis. These concepts are essential to understanding the subsequent chapters and the research conducted in this work.

In Section 2.1, we begin with a description of the multi-stakeholder recommendation setting, including its economic foundations and relationship to fairness in machine learning. We provide in Section 2.2 the preliminaries related to recommender systems, and then specifically as it relates to research in fair recommendation. In Section 2.3, we outline key concepts in decision-focused learning. Finally, in Section 2.4, we provide a review of the existing literature and research studies that have contributed to the advancements in this field.

2.1 Multi-Stakeholder Recommender Systems

A multi-stakeholder recommender system is a recommender system that takes into account the preferences, needs, and constraints of multiple stakeholders involved in the recommendation process. It aims to provide personalised and relevant recommendations while considering the interests of various parties. In this section, we outline the economic foundations of multi-stakeholder recommender systems as they relate to multi-sided platforms, and discuss the relationship between multi-stakeholder objectives and fairness.

2.2.1 Economic Foundations

The economic foundation of multi-sided platforms lies in their ability to facilitate interactions and transactions between multiple distinct groups of users, creating value through network effects and by creating positive externalities that reduce transaction costs [31]. The study of the multi-sided business model was crystallised in the work of Rochet and Tirole on what they termed “two-sided markets” [67]. As highlighted by Abdollahpouri et al. [2], the current understanding among economists is that these markets are frequently characterised by multiple sides rather than just two, and this “multisided-ness” applies specifically to certain business

platforms rather than the entire market [30]. Notably, whereas the traditional business model in economics centred around a firm's ability to produce products for profit, this model was inadequate to explain online businesses such as search engines that were offering their products for free [2].

Multi-sided platforms act as intermediaries, coordinating transactions and interactions between market participants. By reducing search costs, facilitating information exchange, and providing a centralised platform for transactions, these platforms enhance market efficiency [30]. These platforms additionally benefit from network effects, where the value of the platform increases as more users and stakeholders join and interact. The positive externalities brought on by this business model, such as increased user base, improved matching and search efficiency, enhanced trust and reputation, and access to a wider range of products or services, create a virtuous cycle, attracting more users and stakeholders to the platform, which further increases the platform's value and attractiveness [6, 21, 77].

Multi-sided platforms leverage user data to personalise recommendations, advertisements, and user experiences [82]. By analysing user behaviour and interactions, platforms can deliver targeted and relevant content, improving user satisfaction and engagement. Personalisation plays a crucial role in enhancing user engagement and retention on these platforms. By providing personalised and relevant recommendations, the system increases user satisfaction, encourages longer user sessions, and promotes repeat visits [35, 73]. Higher user engagement and retention contribute to the economic success of the platform by attracting advertisers, content providers, and other stakeholders who benefit from increased user attention and interactions [39].

The ability to attract and retain participants from all sides of the business is a crucial factor in the success of a multi-sided platform. Therefore, developers of such platforms have a strong incentive to model and assess the usefulness of the system for all stakeholders involved [2, 97].

Research interest in multi-stakeholder recommendation arises from this economic context. Designing and implementing a multi-stakeholder recommender system can be challenging due to the complexity of balancing the diverse objectives and constraints of stakeholders. It requires

careful consideration of the tradeoffs, effective modelling techniques, and appropriate algorithms to provide recommendations that meet the needs of all stakeholders involved [82].

2.2.2 Relationship to Fairness

Multi-stakeholder recommendation is an expansion of recent endeavours to broaden the factors considered in the evaluation of recommender systems beyond simple accuracy measurements [2, 17]. In recent research, there has been a significant focus on incorporating additional objectives such as diversity, novelty, and long-tail promotion into the generation and evaluation of recommendations [2, 47]. Fairness is an example of a consideration that lies outside the strict optimization of an individual user's personalised results. The question of fairness has been explored from the perspective of multi-stakeholder recommendation under the concept of "multi-sided fairness" [16]. This notion of fairness captures the inherent tradeoff that exists between what may be a fair recommendation for users and what is fair to providers [124].

The multi-stakeholder setting can benefit from advancements in the field of fair machine learning, and additionally provides a different perspective on fairness. Therefore, recent research efforts on fairness in recommendation are relevant to this work [99].

2.2 Fairness in Recommendation

In this section, we first outline some preliminaries relevant to recommender systems. We then provide definitions of fairness and discuss the relationship between fairness and recommender systems.

2.2.1 Recommender Systems: Preliminaries

Within this segment, we provide a brief introduction to the fundamentals of recommender systems. A more detailed overview of recommender systems and the state-of-the-art can be found in [5].

2.2.1.1 Candidate Retrieval and Ranking

It is common for recommender system design to follow a cascading approach [44]. In a cascading design, one recommendation technique is employed first to produce a candidate set of users or items and then a second technique refines the recommendation from among the sets [80].

Candidate retrieval involves selecting a subset of items or candidates from a larger pool of items that are potentially relevant to a user [23]. The purpose of candidate retrieval is to narrow down the search space and focus on a smaller set of items that are easier to process. The selected candidates are then passed to subsequent stages, such as ranking or scoring, where more sophisticated, more computationally intensive algorithms are applied to generate the final recommendations [79].

Fairness can be considered both at the candidate retrieval stage or at the ranking stage. However, because the ranking stage determines the final output of the recommender system, we will mostly focus on this stage.

2.2.1.2 Learning to Rank

Learning to rank is an area of machine learning that focuses on developing models and algorithms to automatically learn the ranking of items or documents based on a set of features [51]. The goal of learning to rank is to create a ranking function that can accurately order a list of items according to their relevance for a given query or user profile. While most of the research in learning to rank is focused on information retrieval tasks, ranking is equally relevant to recommender systems, where recommended items must be prioritised and featured within the constraints of limited screen space.

Learning to rank methods can broadly be categorised into three approaches, namely pointwise, pairwise, and listwise methods [53].

Pointwise Approach. In the pointwise approach, each document or item is assigned a ground truth label, indicating the relevance of the item with regards to a specific query or user [53]. Then the learning to rank problem can be solved as a classification problem in the case of binary

labels, or as a regression problem in the case of numerical scores [22]. At inference time, the items are ranked in descending order based on their predicted values, which refers to the probability that the document or item will be relevant, or to a predicted numerical score [99]. Pointwise approaches assume each user-item rating or query-document relevance datum is independent [89].

Pairwise Approach. In the pairwise approach, the learning to rank problem takes the form of a binary classification task. This classifier compares two documents in relation to a query or user and predicts which one is more relevant. Pairwise approaches assume that pairwise comparisons for two items by the same user are independent [89]. The 0-1 loss of this binary classification problem quantifies the number of inversions in the ranking, indicating how many times the predicted order deviates from the ground truth order [99].

Compared to the pointwise approach, pairwise learning to rank is more computationally demanding. Given every pair of documents needs to be compared, its complexity grows quadratically with the number of documents. This can pose challenges when dealing with large datasets [99].

Additionally, using a loss function based on binary classification may not align well with the evaluation measures used to assess the final rankings. Metrics used to evaluate the output of a ranking model typically focus on the ability of the model to accurately model items at the top of the list, and place a higher priority on those rankings compared to low ranking items. This mismatch between the training objective and final evaluation measure can affect the effectiveness of both the pointwise and the pairwise approach in capturing the true relevance of documents [112, 113].

Listwise Approach. In the listwise approach, each query or user is represented by a group of documents or items and their ground truth labels. The listwise approach views the list of item preferences as a whole and treats different users' lists as independent data points [89]. It seeks to directly optimise the value of the evaluation measure averaged across all rankings produced from the training data. However, optimising these evaluation measures can be challenging since many

of them are not continuous with respect to model parameters. To address this issue, methods often employ smooth approximations or bounds to facilitate optimization.

Several notable examples of listwise approaches include SoftRank [117], ListNet [18], LambdaRank [118], and LambdaMART [119]. Other listwise approaches have directly tackled the task of *collaborative ranking* [120], such as SQL-Rank [88] and DeepRank [121], which we will present in more details in the following section.

2.2.1.3 Collaborative Filtering

Collaborative filtering is a technique used to generate recommendations by leveraging the preferences and behaviours of a group of users. To date, it is the most widely used recommendation algorithm in both academia and industry [8]. It is based on the idea that users who have similar tastes or preferences in the past are likely to have similar preferences in the future [66].

Collaborative filtering approaches be categorised into two main types [5]:

> *User-Based Collaborative Filtering.* This method identifies users with similar preferences and generates recommendations based on the items liked or rated by those similar users. It finds users who have rated or interacted with items in a similar way to the target user and recommends items that those similar users have liked but the target user has not yet encountered.
>
> *Item-Based Collaborative Filtering.* In this approach, the focus is on finding items that are similar to the ones the target user has already liked or interacted with. It identifies items that have been rated or interacted with similarly by users and recommends items that are similar to the ones the target user has shown interest in.

Collaborative filtering algorithms typically use a matrix representation of user-item interactions, where each row represents a user, each column represents an item, and the cells contain ratings or indicators of user-item interactions (e.g., likes, views, purchases).

Collaborative Ranking is a class of collaborative filtering algorithms that seeks to predict how a user will rank items. Collaborative ranking algorithms can be differentiated from rating-oriented collaborative filtering by its objective. While standard collaborative filtering tasks try to minimise the rating prediction error, collaborative ranking tasks aim to minimise the rank prediction error. On this axis, collaborative ranking is similar to learning to rank. However, in practice, most learning to rank applications leverage a set of common explicit features, such as terms' frequencies, while there are no such features available to relate user and item entities in collaborative ranking. As such, while collaborative ranking and collaborative filtering differ in their training objectives, modelling techniques that are used in collaborative filtering are more useful in representing the collaborative ranking task than models traditionally used in learning to rank [114].

The most well known model-based collaborative filtering techniques are matrix factorization, neural collaborative filtering, and singular value decomposition. Matrix factorization in particular became widely known after the Netflix prize in 2006 [48] and is frequently used in academia and the industry. These modelling techniques have also been used for collaborative ranking, for instance matrix factorization in [88] and neural collaborative filtering in [121].

Collaborative filtering has several advantages, including its independence from user or item attributes and content, where data can be more difficult to collect. Collaborative filtering also has some limitations, such as the cold-start problem (difficulty in providing recommendations for new users or items) and the sparsity of user-item interactions in large corpuses [12, 40]. However, these downsides will not be extensively explored in this thesis.

2.2.1.4 Learning with Implicit Feedback

Recommendation and ranking problems typically deal with two kinds of feedback: explicit feedback and implicit feedback [66]. Explicit feedback is direct and explicit input from users regarding their preferences or opinions. Explicit feedback can be in the form of numerical ratings (e.g., giving a movie a rating of 4 out of 5 stars) or textual reviews expressing opinions about the item [109]. Implicit feedback, on the other hand, is derived from user interactions with the

system, such as clicks, purchase history, browsing patterns, time spent on certain items, or even mouse movements.

Explicit feedback can be a more reliable measure of users' ordinal preferences, but is rarely broadly available in real-world settings. Eliciting relevance annotations from experts is infeasible or impossible for many recommendation and ranking applications. Thus, implicit feedback from user behaviour is an attractive source of data in cases where explicit feedback cannot be feasibly collected [45].

2.2.2 Definitions of Fairness

Having provided preliminaries related to recommender systems, we now shift our focus to fair recommendation. We start by providing key definitions of fairness. These definitions are important in order to understand the fairness metrics chosen for a particular machine learning application, as we do in Chapter 3.

Fairness metrics can be defined according to different dimensions. In this section, we take inspiration from the survey completed in [81] and focus on defining fairness around the following aspects: target, subject, granularity, and optimization objective.

Group vs. Individual Fairness. Based on whether the target is to ensure more equitable outcomes at the group-level or individual-level, fairness can be further categorised into group fairness and individual fairness.

> *Group fairness* holds that outcomes should be fair among different groups. There are various ways to divide groups, with the most common approach in the fairness literature being to divide groups based on some explicit fairness-related dimension or protected attribute, such as gender, age, and race. When there are multiple fairness-related attributes, the whole may be divided into numerous subgroups.
>
> *Individual fairness* refers to the idea that similar individuals should be treated similarly. Individual fairness can be theoretically regarded as a special case of group fairness, in which each individual belongs to a unique group.

Subject. The chosen fairness metric will additionally depend on the subject receiving allocation in the recommendation process. Subjects in recommendation can be divided into item fairness, user fairness, and joint fairness.

> *Item fairness* concerns whether the recommendation treats items fairly, such as similar prediction errors for ratings of different types of items or allocating exposure to each item proportional to its relevance.
>
> *User fairness* concerns whether the recommendation is fair to different users, such as similar accuracy for different groups of users or similar recommendation explainability across different users.
>
> *Joint fairness* concerns whether both users and items are treated fairly.

In multi-stakeholder recommendation, unless the provider is the object of the recommendation (for instance, recommending artists on a streaming platform), their fairness concerns tend to be grouped under the category of item fairness.

Granularity. The granularity of the fairness criterion determines the level at which fairness is enforced and can be divided into single fairness and amortised fairness.

> *Single fairness* requires that the recommender system meets fairness requirements each time it generates a single recommendation list.
>
> *Amortised fairness* requires that the cumulative effect of multiple recommendation lists is fair, while a single recommendation list in them may be unfair.

Optimization Objective. The final fairness dimension we will outline is the optimization object and can be divided into treatment-based fairness and impact-based fairness.

> *Treatment-based fairness* only considers whether the treatments of the recommender system are fair or not, such as the predicted scores to different users and the allocated exposure to different items.

Impact-based fairness takes the impact caused by recommendations (i.e., user feedback) into account.

Appendix A contains a lookup table, extending the work conducted in [81], that includes key works belonging to the fair recommendation literature (which will be discussed in Section 2.4) along with their fairness category and nomenclature.

2.2.3 Fairness Criterion

While the fairness definitions in the previous subsection outline some of the different dimensions involved when considering how to approach fairness for a given application, fairness criterions provide a mathematical definition of fairness that is measurable. This section outlines the fairness criteria commonly used in the fair machine learning literature and in fair recommendation and ranking.

Note that these criteria are subjective, and encode normative judgements that are application specific. For more comprehensive insights into the normative values associated with various technical choices in recommendation and ranking, we direct the reader to reference [95].

Equalised Odds. The Equalised Odds criterion [110] focuses on eliminating discriminatory behaviour in machine learning algorithms by ensuring equal predictive accuracy among different groups. This criterion requires that the *False Positive Rates* and *False Negative Rates* are equalised across groups. By achieving equalised odds, the algorithm ensures that individuals from different backgrounds are treated fairly and receive accurate predictions, regardless of their protected attributes or group membership.

A predictor $\widehat{Y}$ satisfies equalised odds with respect to protected attribute A and outcome Y, if $\widehat{Y}$ and A are independent conditional on Y [110].

$$P\{\widehat{Y} = 1 \mid Y = y, A = 0\} = P\{\widehat{Y} = 1 \mid Y = y, A = 1\}, y \in \{0, 1\}$$

Equal Opportunity. The Equal Opportunity criterion [110] seeks to address disparate impact by providing equal opportunity for positive outcomes among different groups by ensuring that

individuals are not unfairly disadvantaged or advantaged based on their membership to a group or protected attributes. This means that individuals within each group should have an equal chance of being correctly identified as positive cases.

A predictor $\widehat{Y}$ satisfies equal opportunity with respect to A and Y if:

$$P\{\widehat{Y} = 1 \mid A = 0, Y = 1\} = P\{\widehat{Y} = 1 \mid A = 1, Y = 1\}$$

Essentially, an algorithm is considered to be fair under the equal opportunity criterion if its *True Positive Rate* is the same across different protected groups.

Demographic Parity. The Demographic Parity criterion [115] aims to achieve a balanced distribution of predicted outcomes across different groups, irrespective of the outcome's accuracy. This criterion requires that the predicted outcomes are distributed proportionally among the groups, aligning with the demographic makeup of the overall population. By adhering to demographic parity, machine learning algorithms aspire to eliminate biases that may disproportionately impact certain groups and promote fairness in the allocation of opportunities and resources.

Under demographic parity, each protected group has the same probability of being classified with the positive outcome [19]. A predictor $\widehat{Y}$ satisfies demographic parity with respect to A if:

$$P\{\widehat{Y} = 1 \mid A = 0\} = P\{\widehat{Y} = 1 \mid A = 1\}$$

Fairness of Exposure/Equity of Attention. In fair recommendation, particularly in fair learning to rank, key consideration is given to the fair allocation of the exposure to the candidate items, given exposure is a limited resource [52]. The exposure based fairness criterion can be defined differently based on which notion of fairness is desired. For instance, a notion of merit (i.e. the relevance of a candidate item to a user) or of demographic parity can be used to determine what constitutes fair allocation of exposure. In all cases, this notion of fairness is defined by quantifying the expected attention a candidate or a group of candidates receives, typically by comparing their average *position bias* [111] to that of other candidates or groups.

Position bias describes the tendency of users to interact with items at the top of a ranked list, which means that the expected exposure of candidates reduces as the ranking position increases. As a result, a slight difference in estimated relevance could result in large differences in exposure between protected groups. Let C correspond to the candidate items to rank and $n = |C|$. The position bias of position j, v_j, is defined as the fraction of users who examine the candidate item at position j [71]. A common measure of positional bias is as follows, where the position bias vector $\mathbf{v}$ is defined with elements $v_j = 1/(1 + j)^p$ for $j \in [n]$ and with $p > 0$ being an arbitrary power [49].

For a given ranking distribution, exposure can be defined as:

$$\text{Exposure}(\boldsymbol{\pi}(i)) = \mathbb{E}_{\boldsymbol{\pi}\sim\Omega}[\mathbf{v}(\boldsymbol{\pi}(i))]$$

where $\boldsymbol{\pi}(i)$ corresponds to the ranking policy. Here, a ranking $\boldsymbol{\pi}$ is a permutation over candidates C, we denote $\boldsymbol{\pi}(i)$ the candidate at rank i, and Ω: $rank(C) \rightarrow [0, 1]$ as the probability mass function over the ranking space. $\mathbf{v}(\boldsymbol{\pi}(i))$ refers to the position bias as defined above.

Individual unfairness in exposure can be expressed as the discrepancy in exposure between two candidates a and b:

$$D(a, b) = |\text{Exposure}(a) - \text{Exposure}(b)|$$

and group fairness can be expressed as the discrepancy in the average exposure between two groups G0 and G1:

$$D(G_0, G_1) = \left| \frac{1}{|G_0|} \sum_{a \in G_0} \text{Exposure}(a) - \frac{1}{|G_1|} \sum_{b \in G_1} \text{Exposure}(b) \right|$$

Note that there is no consensus on the definition of exposure, and while many measures include position bias, not all works agree on its importance [95]. The choice of position bias in defining exposure is motivated by the need to capture the real-world dynamics of ranking, where the position of an item in a ranked list impacts its visibility and subsequent user engagement. Position biases can be estimated with swap experiments [45] or intervention harvesting [103]. In

the simplest of cases, as outlined above, the examination probabilities only depend on the rank of the item with a fixed probability for each rank [60, 116].

2.2.4 Methods for Fair Recommendation

Now that we have outlined fairness considerations in recommender systems, we can formalise the background to the methodologies explored in this thesis. Research in fairness in machine learning has divided fairness methods according to three approaches: pre-processing methods, in-processing methods, and post-processing methods [19].

Pre-processing Methods. One effective approach used to address bias inherent in the data is to preprocess the training data prior to the learning phase. This involves modifying the data to eliminate underlying biases or scrubbing features to remove protected attributes or other identifiers of group membership. Preprocessing methods typically involve altering label values for specific data points or mapping the data to a transformed space [52]. Pre-processing methods are frequently data-oriented approaches that improve fairness by modifying the underlying features or target labels used for training. Compared with other types of methods, there are fewer data-oriented methods in the fair recommendation literature.

In-processing Methods. In-processing methods aim to strike a balance between accuracy and fairness by modifying the learning process itself. These methods typically involve incorporating fairness metrics into the objective function of the primary learning task, either through adversarial learning, regularisation, or reinforcement learning. In-processing methods typically yield better tradeoffs between accuracy and fairness compared to post-processing methods, because finding this balance is at the core of their learning objective [95]. However, it is important to note that this strategy often leads to a non-convex optimization problem, which may not be able to guarantee optimality in the solution. We will proceed to describe popular in-processing methods in the recommendation setting.

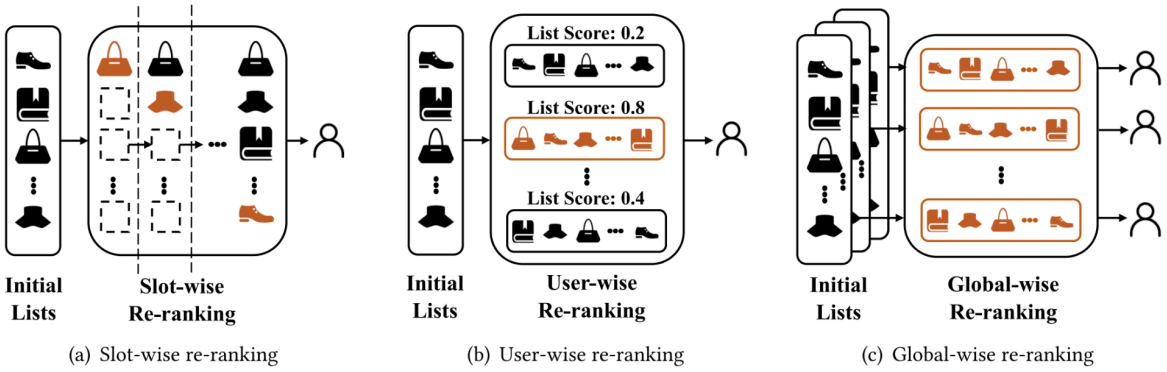
> *Regularisation.* One common approach for improving fairness and reducing bias in recommendations is to add a fairness-related regularisation term to the loss function of a machine learning model. The objective can be maximising the utility under fairness

constraints $L = L_{rec} + \lambda \cdot L_{fair}$, maximising fairness requirements under utility bounds $L = L_{fair} + \lambda \cdot L_{rec}$, or jointly optimising both fairness and utility goals with a reasonable tradeoff [24, 52].

Constrained Optimization. Another approach is to embed constraints into the stochastic policy of a deep learning recommendation model. This can be achieved by including a penalty term for constraint violation in the loss function, as described above, or through reinforcement learning.

Reinforcement Learning. Some works mitigate unfairness in recommendations by introducing information about fairness in the states or rewards of a reinforcement learning algorithm or through additional constraints.

Post-processing Methods. Post-processing methods involve applying transformations to the output scores of a trained model. These transformations are typically carried out by reassigning the predictions assigned by the base models. For instance, this can be done by recomputing the scores or re-ranking the output lists. Post-processing methods usually treat the base model as a black-box and provide model-agnostic flexibility [52]. Much of the literature on fairness in recommender systems has focused on developing post-processing approaches that involve re-ranking the output after the scores or rankings have been produced [95]. Re-ranking approaches can be further categorised under three types [81]. Slot-wise re-ranking ranks a recommendation list by adding items to empty slots in a list one-by-one. User-wise re-ranking tries to directly find the best recommendation list for a user based on the optimization goals of the whole list. Finally, global-wise re-ranking involves re-ranking multiple lists for multiple users simultaneously. The differences between each type is illustrated in Figure 1.



(a) Slot-wise re-ranking (b) User-wise re-ranking (c) Global-wise re-ranking

Figure 1: Different re-ranking types for post-processing recommendations, taken from [81].

Mathematical Programming. Constrained optimization is not only used as an in-processing technique, but is also frequently used for re-ranking through the use of mathematical programming. Considering the focus on re-ranking, methods often take the form of integer linear programs. Stochastic ranking can be used to formulate the problem as a continuous one, making the ranking feasible.

Re-Ranking Algorithms. An algorithmic approach to ranking may involve a greedy strategy with additional requirements to ensure providers receive fair exposure, or allocate exposure in a round-robin manner based on some priority queue until all fairness requirements are satisfied in the system. These algorithmic approaches are often more efficient than mathematical programming.

The methods outlined in this section will be covered in more details in Section 2.4, where we will situate them within the broader literature.

2.3 Decision-Focused Learning

Having presented an introduction to multi-stakeholder recommender systems and the primary motivations driving their exploration, along with an overview of the considerations and methods typically employed in this area, we now shift our focus to the central method examined in this thesis: decision-focused learning.

Decision-focused learning has proven beneficial in many practical applications that involve constrained optimization on predictions. Various domains, including ranking [11], top-k selection [57], and diverse bipartite matching [84] have benefited from decision-focused learning techniques [56]. Given the prevalence of constraint-based approaches in fair recommendation research, as outlined in the previous sections, it is reasonable to explore the application of decision-focused learning as an *in-processing method* for fair recommendation. In this section, we introduce the foundational ideas behind decision-focused learning and techniques relevant to this domain.

Born out of the field of operations research, decision-focused learning seeks to address the challenge that comes with applying constrained optimization programs to the predicted values produced by a machine learning model.

In many real-world applications, the parameters of a constrained optimization problem are uncertain and need to be estimated using data [84]. Typically, these components are tackled separately. First, a machine learning model is trained via a measure of predictive accuracy (e.g. cross-entropy loss, mean squared error, hinge loss) to produce predictions. Second, these predictions are fed into a constrained optimization program to produce a decision. This learning paradigm is called *predict-then-optimise* or *two-stage learning* in decision-focused learning research. The underlying deficiency with this paradigm is that common measures of predictive accuracy, which we can term prediction losses, may not be entirely aligned with the downstream optimization task. Since the machine learning model is trained without knowledge of the optimization problem, it may produce accurate predictions with respect to its prediction loss, but lead to a suboptimal solution in the final evaluation. Note that, in practice, it is impossible to learn a model with no prediction error on any sample. Therefore, accurately approximating the coefficients of an optimization program may not be sufficient to arrive at the desired solution.

To illustrate this problem, we turn to a compelling and intuitive example from Mandi et al. [56]. Suppose we are solving a simple two-item knapsack problem where the items are known to have unit weight and the knapsack capacity is one. The knapsack problem takes the form of:

$$\mathbf{x}^\star(\mathbf{c}) = \underset{\mathbf{x}\in\{0,1\}}{\operatorname{argmax}} \mathbf{c}^\top \mathbf{x} \;\; \texttt{s.t.} \sum_i x_i \leq \text{Capacity}$$

The ground truth item value c implies the ground-truth solution $x^*(c)$. However, we are assuming that at inference time, we do not have access to c and instead must estimate $\hat{c}$ using a machine learning model. Figure 2 illustrates this example and highlights the potential shortcomings of the *predict-then-optimise* approach.

In the diagram, the point (2.5, 3), marked with ✱ represents the true item values. In this case, $x^*(c)$ corresponds to selecting only the second item (0, 1). On the chart, any prediction that falls within the region shaded in blue leads to this optimal decision.

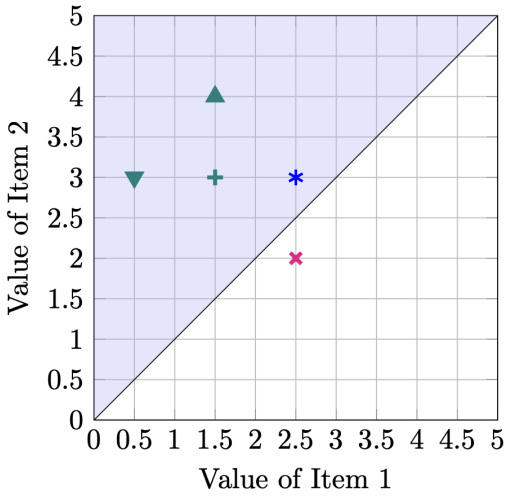


Figure 2: An illustrative example of how two points can have the same prediction error, but lead to different solutions, taken from [56].

Note that although in the above example, the predicted point value (1.5, 3), marked with ✚, and the predicted value (2.5, 2), marked with ✖, are equal distance from the true item value and therefore share the same squared error, they do not lead to the same decision. While the prediction ✚ falls within the optimality region, the point estimate ✖ induces the wrong decision, that is, (1, 0) instead of (0, 1). Further note, that although the point estimates (1.5, 4), marked

with ▲, and (0.5, 3), marked with ▼, have higher squared errors, both lead to the optimal solution from the standpoint of the optimization task.

This insightful example by Mandi et al. demonstrates that the tradeoff between prediction errors across different samples may not perfectly align with the ultimate objectives of the system.

The decision-focused learning paradigm, in contrast to *predict-then-optimise*, seeks to avoid any misalignment between the prediction phase and the optimization phase by training the machine learning model with a loss function that incorporates some knowledge of the error induced by the prediction on the downstream decision. Figure 3 provides an illustration of this end-to-end process.

Recall that the classical approach to this problem within the *predict-then-optimise* paradigm is to use a task agnostic loss function for model training and then use the learned model to solve the optimization problem. Let z correspond to a feature vector that is correlated with the decision variable c from our decision problem, which we wish to learn. In supervised learning, we observe training instances $(z_i, c_i)_{i=1}^N$, where z and c are drawn iid from a joint distribution P, and the goal is to use the training data to learn a model m that minimises the prediction error with respect to the ground truth parameters. The model will have its own parameterization $m(z, \omega)$, where ω represents the model weights. Let $\mathcal{L}$ correspond to the task loss, then the two-stage approach will first solve the problem $min_\omega \, \mathbb{E}_{z,c\sim P} \, [\mathcal{L}(c, m(z, \omega))]$ where the loss function measures the overall accuracy of the prediction without specifically considering how the predicted values will perform when used in the downstream decision problem.

Under the decision-focused paradigm, the model maintains the same parameterization, and we still obtain our predictions $\hat{c} = m(z, \omega)$. However, instead of finding a model that minimises the prediction error, we find a model that minimises the objective function of the optimization task. Defining $x^*(\hat{c}) = \text{argmin}_{x\in\mathcal{F}} f(x, \hat{c})$ to be the optimal decision x for a given coefficient c, the end goal of the decision-focused model is to solve:

$$\mathop{\mathbb{E}}_{z,c\sim P}[\mathcal{L}(x^*(m(z, \omega)), c)]$$

where the loss function measures the end-to-end optimization task loss.

In order to minimise the task loss by gradient descent, the partial derivative of the loss with respect to the prediction model parameters ω must be computed. Since the optimization task loss $\mathcal{L}$ is a function of $x^*(\hat{c})$, the gradient of $\mathcal{L}$ with respect to ω can be expressed by using the chain rule as:

$$\frac{d\mathcal{L}(x^*(\hat{c}), c)}{d\omega} = \frac{d\mathcal{L}(x^*(\hat{c}), c)}{dx^*(\hat{c})} \frac{dx^*(\hat{c})}{d\hat{c}} \frac{d\hat{c}}{d\omega}$$

While the first term and the last term should appear familiar, as they represent the gradient of the model's prediction with respect to its own internal parameterization, and the gradient of the objective with respect to the decision variable, respectively; the middle term is unique to decision-focused learning. The middle term measures how the optimal decision changes with respect to the prediction. Calculating this term represents a core challenge in decision-focused learning. Next we will present two approaches to decision-focus learning, one that obtains this term by differentiating the KKT conditions and another that bypasses it altogether, and discuss their relevance to our fair recommendation setting.

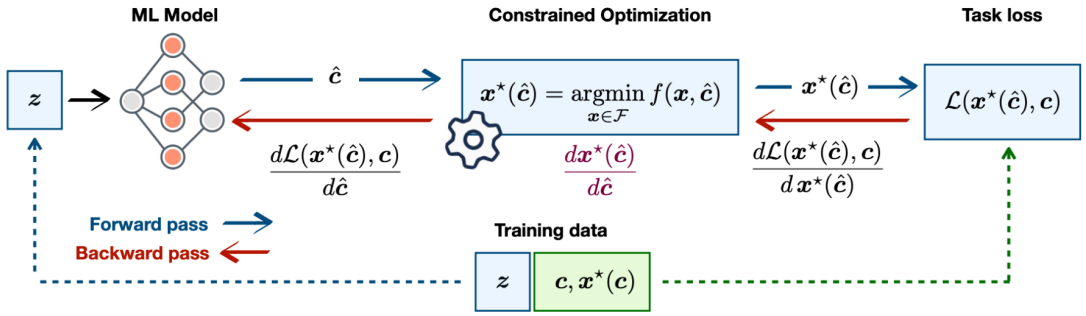


Figure 3: Decision-Focused Learning machine learning pipeline, taken from [56].

2.3.1 OptNet

OptNet is a framework by Amos and Kolter [122] that formulates an optimization problem as a differentiable function, which they then embed as a layer in a neural network model. Since

OptNet is trained using a combination of supervised learning and backpropagation, they need to compute the derivative $\frac{dx^*(c)}{d\hat{c}}$. To obtain this term, they differentiate through the Karush–Kuhn–Tucker conditions.

The Karush–Kuhn–Tucker (KKT) [123] conditions are a set of equations expressing sufficient and necessary conditions for optimality based on the gradients of the objective and constraints around the optimal point [84]. Amos and Kolter differentiate the solution to this linear system by applying the implicit function theorem, whereby the exact gradient $\frac{dx^*(c)}{d\hat{c}}$ can be retrieved by solving the differentiated KKT conditions

$$\begin{bmatrix} \nabla_x^2 f(x,\theta) & A^T \\ diag(\lambda)A & diag(Ax-b) \end{bmatrix} \begin{bmatrix} \frac{dx}{d\theta} \\ \frac{d\lambda}{d\theta} \end{bmatrix} = \begin{bmatrix} \frac{d\nabla_x f(x,\theta)}{d\theta} \\ 0 \end{bmatrix}$$

A continuous problem can be represented via linear equalities as the set $\{x\colon Ax \leq b\}$ for the matrix of constraint coefficients A and constant vector b. Here, (x, λ) is the pair of primal and dual variables that satisfy the KKT conditions.

Wilder et al. later show that this approach not only works for continuous problem formulations, but any continuous relaxation of an integer linear program.

2.3.2 Smart "Predict, Then Optimise"

Smart "Predict, Then Optimise" (SPO), introduced by Elmachtoub and Grigas [28], is another well known approach in decision-focused learning that bypasses the the need to compute the middle term $\frac{dx^*(c)}{d\hat{c}}$ in the backward pass entirely.

Recall that in decision-focused learning, the machine learning model is trained to optimise a criteria that measures the quality of the resulting decisions. From this point of view, generating the parameters $\hat{c}$ is an intermediary step in an end-to-end pipeline. Given the goal is to generate predictions $\hat{c}$ with an optimal solution $x^*(\hat{c})$ whose objective value comes close to the true objective value $x^*(c)$, we can develop a task loss that accounts for the differences between $x^*(c)$

and $x^*(\hat{c})$. We term this task loss the *regret*, which we define as the difference between the full information optimal objective value and the objective value realised using the prediction. In effect, the regret measures the degree of suboptimality of the decision $x^*(\hat{c})$ with respect to the optimal solution $x^*(c)$:

$$Regret(\mathbf{x}^\star(\hat{\mathbf{c}}), \mathbf{c}) = f(\mathbf{x}^\star(\hat{\mathbf{c}}), \mathbf{c}) - f(\mathbf{x}^\star(\mathbf{c}), \mathbf{c})$$

Note that minimising the regret is equivalent to minimising the value of $f(x^*(\hat{c}),\ c)$, $\mathcal{L}(x^*(\hat{c}),\ c)$ in Figure 3, since the term $f(x^*(c),\ c)$ is constant with respect to the prediction model.

Since the gradient of the regret with respect to the cost vector $\hat{\mathbf{c}}$ is zero almost everywhere, SPO uses a surrogate loss function that has subgradients which are useful in training. They start by proposing a convex surrogate upper bound of regret, which they call SPO+ loss:

$$\mathcal{L}_{SPO+}(\mathbf{x}^\star(\hat{\mathbf{c}})) = 2\hat{\mathbf{c}}^\top \mathbf{x}^\star(\mathbf{c}) - \mathbf{c}^\top \mathbf{x}^\star(\mathbf{c}) + \max_{\mathbf{x}\in\mathcal{F}} \{\mathbf{c}^\top \mathbf{x} - 2\hat{\mathbf{c}}^\top \mathbf{x}\}$$

then derive the following subgradient:

$$\mathbf{x}^\star(\mathbf{c}) - \mathbf{x}^\star(2\hat{\mathbf{c}} - \mathbf{c}) \in \partial \mathcal{L}_{SPO+}$$

This subgradient is used to update the model parameters in the backward pass, which circumvents the need to compute $\frac{dx^*(c)}{d\hat{c}}$ [56]. Theoretically, the SPO+ loss exhibits the Fisher consistency property with respect to the regret under specific distributional assumptions. A surrogate loss function demonstrates Fisher consistency when the function that minimises the surrogate loss also minimises the true loss in expectation. This means that minimising the SPO+ loss corresponds to minimising the regret in expectation. Liu and Grigas [125] develop risk bounds for SPO+ loss and show that low excess SPO+ loss risk translates to low excess regret risk, and El Balghiti et al. [27] develop worst case generalisation bounds of the SPO loss. The

SPO framework is applicable to any constrained optimization problem where the cost parameters c appear linearly in the objective function.

Not only has SPO proved to be a very performant decision-focused method in solving a wide variety of tasks [56], this framework provides more flexibility than OptNet. The benefit of OptNet is that it can be used to embed optimization at any layer in a neural network, whereas SPO can only be applied in the final step of the training loop.

2.4 Related Works

This section surveys the direction of prior works in order to situate this thesis within the current research landscape. There are many excellent surveys that provide a broad overview on the topics of multi-stakeholder recommendation [2]; fairness in learning to rank [95, 100, 127] and recommendation [24, 47, 81, 126]; and end-to-end learning [56, 128] that can be consulted to supplement the material in this section.

2.4.1 Multi-Stakeholder Recommendation

Much of the research in multi-stakeholder recommendation has focused on formalising the problem and fewer works have focused on developing methods to mitigate it.

For instance, Burke [16] formalises notions of fairness in personalised recommendations by extending ideas of fairness in classification. He introduces the concepts of multi-sided fairness, which is relevant in platforms that serve a matchmaking function, and provider-side fairness, which is relevant in marketplaces and content sharing platforms. Chakraborty and Biega [104], propose a framework to think about fairness in the matching mechanisms of online sharing economy platforms. Gummadi and Chakraborty [75] experiment with various optimization problems and heuristics to explore the means of achieving two-sided fairness in a ride-hailing service.

Amongst the few method oriented studies on multi-stakeholder recommendation, Sürer et al. [76] propose a post-processing method that incorporates provider-side fairness constraints in a binary integer optimization program. Their provider-side constraint is based on ensuring an amortised

minimum exposure of all providers in the recommendation sets. Patro et al. [63] model the fair recommendation problem as an allocation problem and propose a post-processing algorithm, FairRec, based on a greedy strategy to ensure providers receive fair exposure and customers receive fair recommendation quality. They introduce the fairness metric *Fraction of Satisfied Producer* where a producer is satisfied if their amortised exposure meets a minimum exposure guarantee. This metric is similar to the fairness metric formulated by Sürer et al. [127]. Wu et al. [90] propose a different post-processing algorithm, inspired by Patro et al., where fairness is defined as ensuring the same exposure for all producers and the same utility for all consumers involved. Their definition of fairness in provider exposure is taken from Biega et al. [9].

2.4.2 Fair Recommendation and Ranking

There have been numerous works on fairness in machine learning, including in classification [4, 105, 106] and regression [107, 108]. These works are relevant to the recommendation setting, where recommendation tasks are often formulated as a classification or regression problem, whereas the learning to rank setting requires special considerations [95]. While supervised learning to rank tasks share similarities with classification tasks, there is a fundamental distinction. This brings us back to the distinction between pointwise objectives and ranking objectives, as we previously emphasised when discussing collaborative ranking and collaborative filtering. As a result, ranking, and by extension recommendation, have developed fairness methods specific to their application domain. We present these in a partially chronological order.

The work of Yang and Stoyanovich [129] was one of the first to formalise rank-aware fairness [95]. They study the ranking problem as an unconstrained multi-objective optimization problem and propose definitions and methods that minimise the difference in the representation between groups in a prefix of the ranking. Celis et al. then frame the re-ranking problem as an integer constrained optimization problem [20]. Zehlike et al. [94] build on the work of Yang and Stoyanovich and Celis et al. and propose FA*IR, an algorithmic approach to re-ranking that uses a priority queue to rank groups. These works seek to enforce a proportional allocation of exposure between groups based on notions of demographic parity. [71] This is achieved by

reducing the occurrences of different groups on a subset of the ranking [129] or by placing a limit on the number of items from each group in the top-k positions [20, 94].

Works by Biega et al. and Singh and Joachims explore ideas of how position bias influences fairness of exposure. They argue that exposure should be allocated to individual items or groups of items based on their merit rather than purely based on group size [9, 70]. Biega et al. [9] implement their definition of fairness into a constrained optimization problem, which they formulate into an integer linear program, in which unfairness is minimised subject to constraints on the maximum NDCG-loss. In contrast, Singh and Joachims [70] construct a stochastic ranking policy using the Birkhoff-von Neumann decomposition and formulate a constrained linear optimization problem that can be solved using interior point methods.

Parallel to these works, research on the popularity bias in recommendation explores notions of fairness of exposure from the perspective of improving the exposure coverage of unpopular items (the long-tail) in recommendation lists [1, 130]. Yao and Huang [92] are the first to study fairness in collaborative filtering recommender systems. They introduce four new metrics that address different forms of unfairness. They then incorporate these measures into the loss function using regularisation terms. However, their fairness metrics focus on user-side fairness, not item-side or provider-side fairness. Following their example, Burke et al. [15] further study adding fairness regularisation, but for a regression task using SLIM.

While earlier works on fairness in ranking focused on providing definitions of fairness and developing post-processing methods for re-ranking that incorporated fairness constraints, Zehlike and Castillo [93] were the first to propose an in-processing approach to fair ranking. They develop a modelling approach, DELTR, based on ListNet that incorporates an exposure-based regularisation term into the ranking loss. Stronger results were reported by Singh and Joachims [71] with their approach, Fair-PG-Rank, which similarly leverages the formulation from ListNet, but learns fair ranking policies using a policy gradient method.

The work of Beutel et al. [7] focus on establishing fairness in recommendation through pairwise comparisons. Their work is based on that of Yao and Huang [92], and try to improve fairness in the resulting ranking through pairwise learning. They add a fairness regulariser based on

randomised experiments. The authors additionally highlight a limitation in applying existing fair learning to rank methods to recommendation, mainly that these work focused on un-personalised rankings where relevancy is known for all items, and data sparsity and bias are not as salient issues.

This is acknowledged in the work of Yadav et al. [91], where they modify the method of Fair-PG-Rank to work in an implicit setting. Specifically, they define a class of amortised fairness-of-exposure constraints that can be chosen based on the needs of an application, and show how these fairness criteria can be enforced despite the selection biases in implicit feedback data. They term their new method FULTR, and show that their proposed algorithm can learn accurate and fair ranking policies from biassed and noisy feedback. However, their problem and dataset are still situated within research in information retrieval.

Taking inspiration from the works of Yadav et al. [91] and Singh and Joachims [70], Kotary et al. [49] introduce *Smart Predict and Optimise for Fair Ranking (SPOFR)*, an integrated optimization and learning framework for fairness-constrained learning to rank. Their work responds to one of the challenges with regularisation-based approaches, such as DELTR and FULTR, when it comes to finding the parameter λ. This value must be treated as a hyperparameter, increasing the computational effort required to find desirable solutions. Moreover, when a tradeoff between fairness and utility is desired, it cannot be controlled by specifying an allowable magnitude for fairness violation because of the lack of a reliable relationship between the hyperparameter λ and the resulting constraint violations. In contrast, SPOFR's end-to-end framework includes a constrained optimization sub-model and produces ranking policies that are guaranteed to satisfy fairness constraints, while allowing for fine control of the fairness-utility tradeoff.

From these works, stochastic ranking has emerged as a standard approach to mitigate exposure inequality in ranking [25, 26, 32, 36, 42, 49, 62, 70, 72, 78, 87]. Stochastic ranking is not without criticism. In one-shot settings, or settings where it is necessary to respect fairness constraints in each output ranking (rather than achieving fairness in expectation) [38], stochastic rankings fail to meet this condition. Further, Bower et al. [13] argue that when fairness is only enforced at the

ranking stage of a recommendation and not at the candidate stage, that stochastic ranking policies can exacerbate inequalities that exist in the candidate set. However, other works argue that given that the presence of uncertainty in many ranking applications is itself a source of unfairness, giving different candidates a fair chance is more equitable than any deterministic approach.

An intuitive example was given by the authors of [72] in a comment during the review of their paper [131]. Imagine we have two candidate items with equal relevance to a user that are indistinguishable for practical purposes, but only one position is available in that user's list. The authors ask, what is the fair thing to do? They argue that a randomised outcome is both fair in the one-shot setting and repeated setting. A randomised outcome (such as a coinflip) is fair *ex ante* since each candidate has the same likelihood of being included in the user's list. *Ex post*, this outcome may appear unfair because the two candidates were equally suitable, but only one was chosen for display. However, in a repeated setting, it becomes clear that fairness is also achieved *ex post* since the candidates will obtain their expected share of preferential treatment. While randomisation may not always be appropriate for high-stake one-shot settings — although it can still be beneficial, considering relying on other deterministic methods like alphabetical order to break ties may introduce its own biases — in most multi-sided platforms (e.g. e-commerce, social media, streaming platforms) where the same set of items are displayed and ranked many times and stakes are fairly low, randomisation can contribute to promoting fairer outcomes in the long run [60, 72].

2.4.3 End-to-End Learning

End-to-end learning has many connotations in machine learning. In decision-focused learning, end-to-end learning refers to the integration of optimization layers within a deep learning pipeline [29]. In fair recommendation, end-to-end learning tends to refer to in-processing methods, where fairness is enforced during training, as opposed to a two-step approach where a model is first trained and then fairness is enforced during post-processing [102]. Broadly speaking, end-to-end learning refers to training a single model to perform a task from raw input to final output [83].

The issues highlighted by end-to-end learning often focus on the idea of potential misalignment between the objectives of multiple modelling approaches, and how such misalignment can lead to undesirable outcomes in the system.

While the in-processing examples in Section 2.4.2 can be interpreted as an end-to-end approach to fair ranking, there has also been additional research in developing end-to-end frameworks for supervised gradient based learning of ranking functions [3, 65].

In the field of decision-focused learning, works by Amos and Kolter [122] set the groundwork for using constrained convex optimization as a layer in an end-to-end architecture. Subsequently, Donti et al. [137] propose a *predict-and-optimise* model in which quadratic programs with stochastic constraints were integrated in-the-loop of training to provide accurate solutions to power generator scheduling problems. Concurrently, Elmachtoub and Grigas [28] produce another seminal work presenting an alternative methodology for end-to-end learning with constrained optimization, in which a subgradient calculation is used for the backpropagation of regret loss.

Additionally, Wilder et al. [84] introduce an alternative framework to predict-and-optimise linear programming problems, based on exact differentiation of a smoothed surrogate model [128]. While the work of Wilder et al. mainly focuses on formalising their framework, they experiment on a diverse recommendation task. Specifically, they employ submodular optimization techniques to address a specific problem: selecting a diverse set of items that collectively cover a wide range of topics. Unlike traditional recommendation tasks, the user ratings are already known and do not require prediction. Instead, the learning objective focuses on predicting the association of topics with the items. To evaluate their approach, the authors conduct experiments using the MovieLens dataset.

Wang et al. [138] similarly experiment with a recommendation setting. Their work includes an exploration of decision-focused learning for the broadcasting problem. In this domain, a broadcasting company chooses a subset out of a larger set of available movies to acquire and show to their customers within a budget constraint. This is a variant of the classic facility location problem. In this case, the user's preferences are unknown and need to be predicted. The

goal is to maximise the overall satisfaction of users without exceeding the budget constraint. While their experiment includes methods that are more closely related to our setting, in their case the task is to find a global recommendation set rather than a set of recommendations per user.

Mao et al. [139] apply decision-focused learning, specifically the end-to-end framework by Wilder et al. [84], to the task of inventory prediction and contract allocation for guaranteed delivery advertising. Although their setting shares practical concerns with multi-stakeholder recommendation on multi-sided platforms, their optimization task and modelling approach are significantly different from ours.

Overall, the work that comes closest to ours is by Kotary et al. [49], which we have introduced in the previous section.

Chapter 3 Models

In Chapter 2, we showed how research in fair recommendation led to the development of in-processing fair learning to rank techniques that could be trained in an end-to-end manner. In this chapter, we focus on integrating this work within a recommendation setting by combining these techniques with collaborative filtering.

Section 3.1 introduces our problem formulation and notation, and presents a training objective that incorporates fairness constraints. Section 3.2 presents the neural matrix factorisation model we use in our experiments. Section 3.3 introduces our baseline regularisation-based model; consisting of a stochastic ranking framework using a Plackett-Luce model that is then trained using a policy-gradient approach. Section 3.4 introduces our decision-focused approach, involving a different class of stochastic ranking policies using doubly stochastic matrices and the Birkhoff–von Neumann decomposition, and outlines how this can be integrated with SPO for end-to-end learning. Finally, Section 3.5 proposes how to combine our neural matrix factorisation model with these learning to rank approaches and outlines the utility and fairness metrics used for our training objective.

3.1 Problem Formulation

In this section, we formalise the problem and introduce our notation. The learning to rank task consists in learning a mapping between a list of n items and a permutation (ranking) r of the list, which defines the ordering of the items.

In a traditional learning to rank setting, the ordering is determined in response to a query, q. Instead of a query, a recommender system leverages collaborative filtering to identify the most relevant information for the user. The goal of collaborative filtering is to predict a personalised score $y_{ui} \in Y$ that reflects the preference level of user u towards item i, where $1 \leq u \leq m$ and $1 \leq i \leq n$. The y_{ui} are supervision labels that associate a non-negative value with each item.

In our setting, we deal with implicit feedback data, such that y_{ui}= 1, if an interaction between the user and item is observed; and 0 otherwise.

The predicted value then takes the form of $\hat{y}_{ui}$. In the personalised recommendation setting:

$$\hat{y}_{ui} = f\left(u, i|\mathbf{p}_u, \mathbf{q}_i\right) = \mathbf{p}_u^T \mathbf{q}_i = \sum_{k=1}^{K} p_{uk} q_{ik}$$

We will show how this formulation can be generalised using neural networks in Section 3.2. Each combination of u and i and its corresponding label y represents a unique user-item pair. These user-item pairs can be reformulated as a list $x_u \in X$ of k items sampled from available items to be ranked for each user u. Let $\mathbf{y}_u$represent the vector of relevances scores for the items in the list for user u.

The goal in learning to rank tasks is to predict, for any user, a distribution over rankings π, called a ranking policy, from which individual rankings can be sampled. The utility of a stochastic policy π for a user u is defined as the expectation of a ranking metric Δ over π as:

$$U\left(\pi, u\right) = \mathbb{E}_{r\sim\pi}\left[\Delta\left(r, \boldsymbol{y}_u\right)\right]$$

Let $\boldsymbol{\mathcal{M}}_\theta$ be a machine learning model, with parameters θ, which takes as input a user-items list and returns a ranking policy. The training objective is to find the parameters θ^* that maximise the empirical risk:

$$\theta^* = \underset{\theta}{argmax}\ \frac{1}{N}\sum_{u=1}^{N} U\left(\mathcal{M}_\theta\left(\boldsymbol{x_u}\right), \boldsymbol{y_u}\right)$$

Since we are not interested in merely optimising for this utility measure, as in traditional learning to rank, but wish to take into consideration notions of fairness, we include a constraint in the learning problem that enforces a notion of fair allocation of exposure as in [71]. The resulting constrained empirical risk problem can be formulated as follows:

$$\theta^* = \underset{\theta}{argmax}\ \frac{1}{N}\sum_{u=1}^{N} U\left(\mathcal{M}_\theta\left(\boldsymbol{x_u}\right), \boldsymbol{y_u}\right)$$
$$\texttt{s.t. } \left|\mathrm{v}\left(\mathcal{M}_\theta\left(\boldsymbol{x_u}\right), g\right)\right| \leq \delta \quad \forall u \in [N], g \in \mathcal{G}$$

Here $g \in \boldsymbol{\mathcal{G}}$ corresponds to the group membership of the items to be ranked. The utility measure and fairness constraint will be further formulated in Section 3.5. For now we leave a general notation, and note that the framework only requires that each be formulated as a linear function.

3.2 Latent Factor Models

Amongst latent factor models, matrix factorization is a frequent choice due to its ability to produce useful latent feature representations of users and items [120]. However, the main disadvantage with the standard matrix factorization formulation for our use case is that it is non-differentiable. It will become clearer as we proceed through this chapter why this is an undesirable property for the type of approaches we wish to explore, and why we decided to turn towards a neural network formulation of this model instead.

3.2.1 Deep Learning-Based Matrix Factorization

Neural collaborative filtering combines neural networks with matrix factorization techniques to learn user and item embeddings. It uses only the identity of a user and an item as the input feature, transforming it to a binarized sparse vector with one-hot encoding. The layer above the input layer is the embedding layer, which serves as a fully connected layer responsible for transforming the sparse representation into a dense vector. The resulting embedding can be interpreted as the latent vector representing the user or item within the framework of a latent factor model [41].

The original formulation of the neural matrix factorization architecture is outlined in Figure 4. As shown, this model is structured with two subnetworks made up of generalised matrix factorization (GMF) and a standard multi-layer perceptron (MLP), and replaces the dot product operation by modelling the interaction from these two pathways [96].

The GMF is a generalisation of matrix factorization, where the input is the element wise product of user and item latent factors, which consists of two neural layers:

$$\mathbf{x} = \mathbf{p}_u \odot \mathbf{q}_i$$
$$\hat{y}_{ui} = \alpha(\mathbf{h}^\top \mathbf{x})$$

where $\odot$ denotes the element-wise product of vectors. $\mathbf{P} \in \mathbb{R}^{m \times k}$ and $\mathbf{Q} \in \mathbb{R}^{n \times k}$ correspond to the user and item latent matrix. Here, $\mathbf{p}_u \in \mathbb{R}^k$ and $\mathbf{q}_i \in \mathbb{R}^k$ denote the latent factors for user u and item i, and α and $\mathbf{h}$ denotes the activation function and edge weights of the output layer, respectively. $\hat{y}_{ui}$ is the predicted score that the user u might give to the item i.

The MLP model is then defined as:

$$\mathbf{z}_1 = \phi_1(\mathbf{p}_u, \mathbf{q}_i) = \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix},$$
$$\phi_2(\mathbf{z}_1) = a_2(\mathbf{W}_2^T \mathbf{z}_1 + \mathbf{b}_2),$$
$$\ldots\ldots$$
$$\phi_L(\mathbf{z}_{L-1}) = a_L(\mathbf{W}_L^T \mathbf{z}_{L-1} + \mathbf{b}_L),$$
$$\hat{y}_{ui} = \sigma(\mathbf{h}^T \phi_L(\mathbf{z}_{L-1})),$$

where $\mathbf{W}$, $\mathbf{b}$ and α denote the weight matrix parameterized by θ, the bias vector, and the activation function. ϕ^* denotes the function of the corresponding layer. $\mathbf{z}$ denotes the output.

The neural matrix factorization (NeuMF) layer then concatenates the second to last layer in the two subnetworks to create a feature vector, which is passed to further layers, effectively fusing the results of GMF and MLP:

$$\hat{y}_{ui} = \sigma(\mathbf{h}^T a(\mathbf{p}_u \odot \mathbf{q}_i + \mathbf{W} \begin{bmatrix} \mathbf{p}_u \\ \mathbf{q}_i \end{bmatrix} + \mathbf{b})).$$

The final output of the model is the predicted score $\hat{y}_{ui}$, and training is performed by minimising the pointwise loss between $\hat{y}_{ui}$ and the target value y_{ui}. Although as we will highlight later, our

setting will not only focus on pointwise loss, but also incorporate ranking objectives, we will present the pointwise approach as it represents one of our baselines in our experiments.

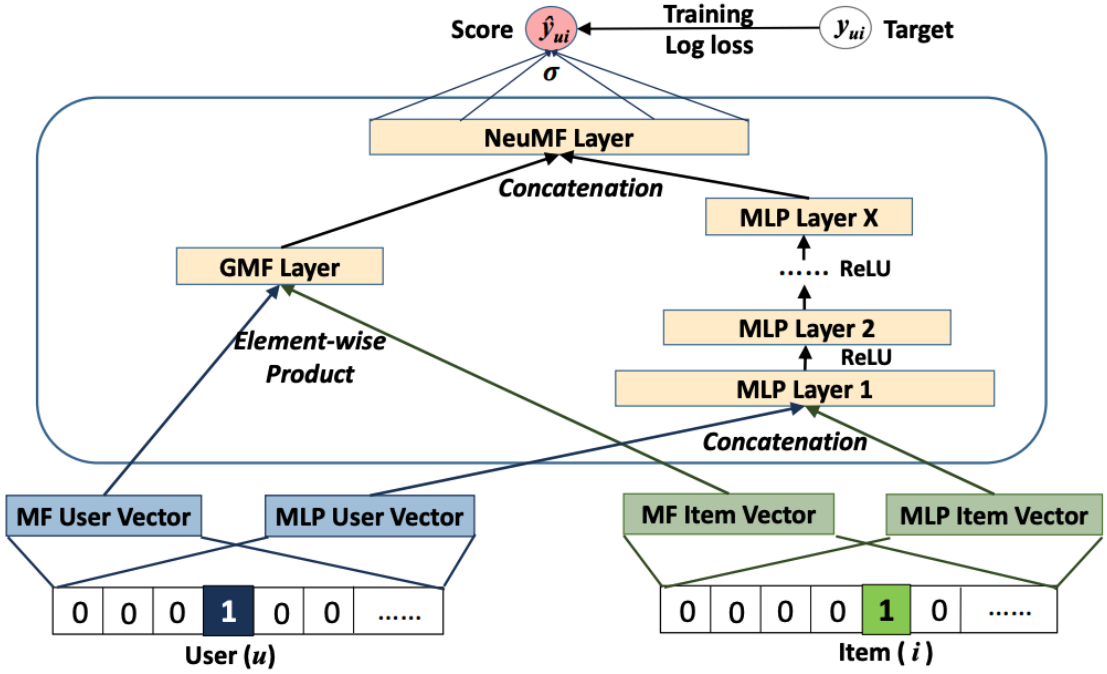


Figure 4: Neural Matrix Factorization architecture, taken from [41].

In order to perform pointwise prediction using implicit data, we need to constrain the prediction score $\hat{y}_{ui}$ to a binary range, which we can achieve by using a probabilistic function, such as the logistic function, as the activation function for the output layer. The likelihood function is defined as follows:

$$p(\mathcal{Y}, \mathcal{Y}^{-} | \mathbf{P}, \mathbf{Q}, \Theta_f) = \prod_{(u,i) \in \mathcal{Y}} \hat{y}_{ui} \prod_{(u,j) \in \mathcal{Y}^{-}} (1 - \hat{y}_{uj}).$$

By taking the negative logarithm of the likelihood function, we obtain:

$$L = -\sum_{(u,i)\in\mathcal{Y}} \log \hat{y}_{ui} - \sum_{(u,j)\in\mathcal{Y}^-} \log(1-\hat{y}_{uj})$$
$$= -\sum_{(u,i)\in\mathcal{Y}\cup\mathcal{Y}^-} y_{ui}\log \hat{y}_{ui} + (1-y_{ui})\log(1-\hat{y}_{ui}).$$

The equation above represents the objective function to minimise during training for the pointwise approach, and optimization can be achieved using stochastic gradient descent. Next, we proceed to show how to define a loss function that not only takes into consideration the ranking objective, but also a fairness component.

3.3 Regularisation-Based Fair Learning to Rank

In this section, we present a state-of-the-art in-processing fair learning to rank technique, which involves embedding a fairness constraint into the stochastic policy of the ranking model using regularisation. This is achieved by including a penalty term for constraint violation in the loss function. This model will be used as a baseline to compare to our decision-focused learning approach in order to evaluate whether the decision-focused approach provides fine control of the fairness/utility trade-off.

3.3.1 Plackett-Luce Ranking Models

The Plackett-Luce (PL) model [54, 64] has often been employed to model a probabilistic distribution over rankings [14, 36, 62, 71, 91, 93]. The PL model model defines a probability distribution on permutations of items, referred to as permutation probabilities. Let π denote a permutation of the items to be ranked and $\pi^{-1}(i)$ denote the item in the i-th position in π [51]. Further suppose that each item is assigned a non-negative score $s = \{s_1, s_2, \dots, s_n\}$.

The PL model defines the probability of permutation π based on scores s as follows:

$$P_s(\pi) = \prod_{i=1}^{n} \frac{s_{\pi^{-1}(i)}}{\sum_{j=i}^{n} s_{\pi^{-1}(j)}}$$

The probabilities of permutations naturally form a probability distribution over the set of permutations, that is, for each π, we have $P_s(\pi)$, and $\sum_{\pi \in \Omega_n} P_s(\pi) = 1$.

The PL formulation has certain desirable properties. For instance, the ranking that sorts the items in descending order according to their score shares the highest probability while the ranking that sorts the items in ascending order has the lower probability. That is, if $s_1 > s_2 > \ldots > s_n$, then $P_s(\langle 1, 2, \ldots, n \rangle)$ has the highest permutation probability and $P_s(\langle n, n-1, \ldots, 1 \rangle)$ has the lowest permutation probability for the items [18]. Additionally, the PL model defines a probability distribution on top-k subgroups, referred to as top-k probability, where n in the above equation is replaced by k.

Given items and permutations of items, we can define a top-k subgroup $g[x_1, \ldots x_k]$ represents all permutations whose top k items are $x_1, \ldots x_k$. The top-k probability of subgroup $g[x_1, \ldots x_k]$ is defined as [51]:

$$P_s(g[x_1 \cdots x_j]) = \prod_{i=1}^{k} \frac{s_{x_i}}{\sum_{j=i}^{n} s_{x_i}}$$

The most cited permutation probability model [88], ListNet, was developed in Cao et al. [18], ListNet makes use of a parameterised PL model:

$$P_{\mathcal{M}_\theta(\mathbf{x})} = \prod_{i=1}^{n} \frac{\exp(\mathcal{M}_\theta(x_i))}{\sum_{j=i}^{n_i} \exp(\mathcal{M}_\theta(x_j))}$$

where $\boldsymbol{\mathcal{M}}_\theta$ is a neural network model with parameters θ and $\boldsymbol{\mathcal{M}}_\theta(\mathbf{x})$ is a list of scores given by the neural network. The top-k probability of the ground truth labels can be calculated similarly, by replacing $\boldsymbol{\mathcal{M}}_\theta(x_i)$ with y_i.

Applying the PL model to our setting leads to the following formulation. Starting with our scoring model $\mathcal{M}_\theta$, which can take the form of any parameterized differentiable machine learning model, and in our case represents our neural matrix factorization model. Given an input representing the feature vectors of all user-item pairs of the candidate set, the scoring model outputs a vector of scores $\mathcal{M}_\theta(\mathbf{x}^u) = (\mathcal{M}_\theta(x_1^u), \mathcal{M}_\theta(x_2^u), \ldots, \mathcal{M}_\theta(x_n^u))$.

$$\pi_\theta(r|u) = \prod_{i=1}^{n_u} \frac{\exp(\mathcal{M}_\theta(x^u_{r(i)}))}{\exp(\mathcal{M}_\theta(x^u_{r(i)})) + \ldots + \exp(\mathcal{M}_\theta(x^u_{r(n_u)}))}$$

To sample a ranking, starting from the top, items are drawn recursively from the probability distribution resulting from the Softmax over the scores of the remaining items in the candidate set, until the set is empty [71].

3.3.2 Policy-Gradient Learning for Fair Learning to Rank

PL ranking models can be optimised via policy-gradients [62, 71, 91]. This approach involves searching the policy space π for a model that maximises the training objective defined in Section 3.1. Using a Lagrange multiplier, this is equivalent to:

$$\hat{\pi}^*_\delta = \text{argmax}_\pi \min_{\lambda \geq 0} \frac{1}{N} \sum_{u=1}^{N} U(\pi|u) - \lambda \left(\frac{1}{N} \sum_{u=1}^{N} \mathcal{D}(\pi|u) - \delta \right)$$

where, $\mathcal{D} = |\mathrm{v}\left(\mathcal{M}_\theta\left(\boldsymbol{x}_{\boldsymbol{q}}\right), g\right)|$.

Considering the permutation space is exponential in *k*, taking the gradient with respect to θ over this expectation is not feasible. In order to overcome this, the log-derivative trick from the *REINFORCE* algorithm [85] can be used as follows:

$$\begin{aligned} \nabla_\theta U(\pi_\theta|u) &= \nabla_\theta \mathbb{E}_{r\sim\pi_\theta(\cdot|u)}[\Delta(r|u)] \\ &= \mathbb{E}_{r\sim\pi_\theta(\cdot|u)}[\nabla_\theta \log \pi_\theta(r|u)\Delta(r|u)] \end{aligned}$$

This policy gradient is composed in two parts: a gradient with respect to the log probability of each sampled rankings multiplied by the reward for that ranking. Note that the log-derivative

trick allows us to rewrite expectations in a way that is amenable to Monte Carlo approximation [46]. The final expectation can be approximated via sampling from the PL model described in the previous section.

The gradient of the fairness disparity can also be formulated in this way:

$$\mathbb{E}_{r\sim\pi_\theta}\left[\left(\sum_{x\in G_0} v_r(x) - \sum_{x\in G_1} v_r(x)\right)\nabla_\theta \log\pi_\theta(r|u)\right]$$

where v corresponds to the position bias vector and x to an item in the ranking. Directly optimising the ranking policy via policy-gradient learning has two advantages over most conventional learning to rank algorithms, which optimise upper bounds or heuristic proxy measures. First, the learning algorithm directly optimises a specified utility metric, which aligns the training objective with the offline evaluation criteria. Second, the same policy-gradient approach can be used to incorporate other objectives, such as fairness of exposure, since it is also quantified in expectation over rankings. Overall, the use of policy-gradient optimization in the space of stochastic ranking policies elegantly handles the non-smoothness inherent in rankings [71].

3.4 Decision-Focused Fair Learning to Rank

In this section we present the core method of this thesis, decision-focused learning for fair learning to rank. This technique involves a different class of stochastic ranking policies based on doubly stochastic matrices and the Birkhoff–von Neumann decomposition. These can then be used to formulate a linear objective in a linear program that is then integrated into the training loop by leveraging SPO for end-to-end learning.

3.4.1 Doubly Stochastic Matrix Ranking Models

While the PL model represents one approach to modelling a probabilistic distribution over rankings, another approach is to model the probability distribution over ranking as a doubly stochastic matrix.

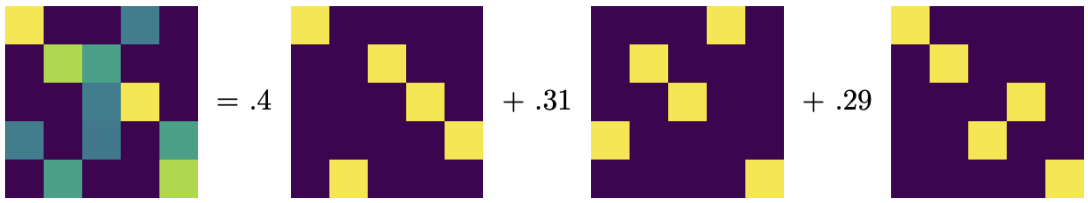


Figure 5: Illustration of the Birkhoff–von Neumann decomposition, taken from [43].

A Doubly Stochastic Matrix (DSM) is a $n \times n$ matrix of non-negative real numbers such that each one of its rows and columns sum to 1 [68]. Using the Birkhoff–von Neumann decomposition [10], any DMS π can be decomposed into a convex combination of at most $(n-1)^2 + 1$ permutation matrices $P^{(i)}$, each associated with a coefficient $\mu_i \leq 0$, which can then represent the probability of sampling each ranking. A ranking policy is inferred from the set of resulting convex coefficients μ_i, forming a discrete probability distribution. Each permutation has a likelihood equal to its respective coefficient [43, 49]. In our case, the permutation matrices correspond to deterministic rankings of the item set and the coefficients correspond to the probability of sampling each ranking:

$$\pi = \sum_{i=1}^{(n-1)^2+1} \mu_i P^{(i)}$$

Any linear utility metric on rankings can be formulated as a linear function on their permutation matrices, which can then be applied to any square matrix. In particular, applying the utility metric operator to a doubly stochastic matrix π results in the expected utility over rankings sampled from its inferred policy. Given item relevance scores $\boldsymbol{y}$:

$$\begin{aligned}\mathbb{E}_{r\sim\pi} U(r, \boldsymbol{y}) &= \sum_{i=1}^{(n-1)^2+1} \mu_i \boldsymbol{y}^\top P^{(i)} \boldsymbol{w} \\ &= \boldsymbol{y}^\top \left(\sum_{i=1}^{(n-1)^2+1} \mu_i P^{(i)} \right) \boldsymbol{w} = \boldsymbol{y}^\top \pi \boldsymbol{w}\end{aligned}$$

where $\boldsymbol{w}$ is a vector of weights. The expected utility of a ranking sampled from a ranking policy π can thus be represented as a linear function on π, which serves as the objective function during training. This analytical evaluation of expected utility is key to optimising fairness-constrained ranking policies in an end-to-end manner [49].

3.4.2 Optimal Fair Ranking Policies

The key problem with optimising over rankings is the combinatorial nature of such problems. By extending the class of rankings to probabilistic rankings, as was demonstrated above, we can formulate the optimization problem as a linear program, which can be enhanced with fairness constraints.

The linear programming model for optimising fair ranking functions, which follows the formulations presented in [70] and [49], can be expressed as:

$$
\begin{aligned}
&\pi^*(\hat{\boldsymbol{y}}_q) = \operatorname{argmax}_\pi \ \hat{\boldsymbol{y}}_q^\top \pi \boldsymbol{w} \\
&\text{s.t.} \sum_j \pi_{ij} = 1 \quad \forall i \in [n] \\
&\sum_i \pi_{ij} = 1 \quad \forall j \in [n] \\
&0 \leq \pi_{ij} \leq 1 \quad \forall i, j \in [n] \\
&\pi \text{ is fair}
\end{aligned}
$$

This optimization program can be solved standalone as a post-processing step or, as we will present next, in an end-to-end manner by incorporating the constrained optimization problem directly into the training loop.

3.4.3 Smart "Predict, Then Optimise" for Fair Learning to Rank

In order to train the fair ranking model in an end-to-end manner, we turn to the formulation outlined in Section 2.3.2 on Smart "Predict, then Optimise" (SPO). Given the ranking objective has been reformulated as a constrained linear program, SPO can now be used to incorporate this optimization problem into the training loop. Kotary et al. [49] provide the only example of

decision-focused learning for fair ranking problems to date, and we will refer mostly to their work in this section.

We begin by formulating the regret between the exact and approximate policies as:

$$\mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \boldsymbol{y}^\top \pi^*(\boldsymbol{y})\boldsymbol{w} - \boldsymbol{y}^\top \pi^*(\hat{\boldsymbol{y}})\boldsymbol{w}$$

The regret measures the loss in objective value, relative to the true cost function, induced by the predicted cost. Following the SPO approach, we can formulate the convex surrogate loss function SPO+, which forms a convex upper-bounding function over $\mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}})$. Its gradient is computed as follows:

$$\frac{\partial}{\partial \boldsymbol{y}} \mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}) \approx \frac{\partial}{\partial \boldsymbol{y}} \mathcal{L}_{\mathsf{SPO+}}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \pi^*(2\hat{\boldsymbol{y}} - \boldsymbol{y}) - \pi^*(\boldsymbol{y})$$

By definition, $y^\top \pi^*(y) \geq y^\top \pi^*(\hat{y})$ and therefore $\mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}) \geq 0$. Thus, finding the $\hat{y}$ minimising $\mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}})$ is equivalent to finding the $\hat{y}$ maximising $y^\top \pi^*(\hat{y})$, since $y^\top \pi^*(y)$ is a constant value. The goal is to maximise the empirical utility, equal to $y^\top \pi^*(y)w$ for ground truth relevance score y. A vectorized form can be written as:

$$\boldsymbol{y}^\top \pi \boldsymbol{w} = \overrightarrow{(\boldsymbol{y}^\top \boldsymbol{w})} \cdot \vec{\pi}$$

where $\overline{A}$ represents the row-major-order vectorization of a matrix A. Then, the regret induced by the prediction of the cost coefficient $\hat{y}$ is:

$$\mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \overrightarrow{(\boldsymbol{y}^\top \boldsymbol{w})} \cdot \overrightarrow{\pi^*(\boldsymbol{y})} - \overrightarrow{(\boldsymbol{y}^\top \boldsymbol{w})} \cdot \overrightarrow{\pi^*(\hat{\boldsymbol{y}})}$$

and gradient can be approximated as:

$$\overrightarrow{\pi^* \left(2\hat{\boldsymbol{y}}^\top \boldsymbol{w} - \boldsymbol{y}^\top \boldsymbol{w}\right)} - \overrightarrow{\pi^* \left(\boldsymbol{y}^\top \boldsymbol{w}\right)}$$

The resulting framework operates as follows. First, for a given user u and associated item list x_u, a neural network model $\boldsymbol{\mathcal{M}}_\theta$ is used to predict the relevance scores $\mathbf{y}_u$. Second, the predicted scores are used to specify the objective function of the linear program, which when solved, returns a fair and optimal policy with respect to the predicted values, $\pi^*(\hat{y}_u)$. Then, we solve for the objective value with respect to the ground truth values $\pi^*(y_u)$, which additionally corresponds to the ground truth optimal policy. Next, we compute the regret function, $\pi^*(y_u)$ - $\pi^*(\hat{y}_u)$, which measures the loss of optimality relative to the true optimal policy. Finally, we compute the gradient of the SPO loss $\theta - \alpha\nabla\mathcal{L}(\boldsymbol{y}^\top\boldsymbol{w}, \hat{\boldsymbol{y}}^\top\boldsymbol{y})\frac{\partial\hat{\boldsymbol{y}}^\top\boldsymbol{w}}{\partial\theta}$ and update the weights via the chain rule using automatic differentiation.

3.5 Combining Neural Matrix Factorization and Fair Learning to Rank

Now that we have introduced the problem formulation, as well as the techniques explored as part of this thesis, we can describe how we integrate each method to create the models used in our experiments. The model closest to the formulation outlined in Section 3.3 is the FULTR model by Yadav et. al. [91], as they develop a version of FULTR that is parameterised using an MLP model, which brings it closer to our setting. In contrast, we only have one model that fits the decision-focused setting, SPOFR by Kotary et. al. [49], which is also parameterised using an MLP model. In each case, we replace the MLP model $\boldsymbol{\mathcal{M}}_\theta$ with our Neural Matrix Factorisation model (NeuMF) described in Section 3.2. To do so, we replace the pointwise loss described in 3.2.1 with our training objective from Section 3.1, that is:

$$\theta^* = \underset{\theta}{argmax}\ \frac{1}{N}\sum_{u=1}^{N} U\left(\mathcal{M}_\theta\left(\boldsymbol{x_u}\right), \boldsymbol{y_u}\right)$$
$$\texttt{s.t.}\ \left|\mathrm{v}\left(\mathcal{M}_\theta\left(\boldsymbol{x_u}\right), g\right)\right| \leq \delta \quad \forall u \in [N], g \in \mathcal{G}$$

We refer to our NeuMF models with modified objectives as NeuMF-FULTR and NeuMF-SPOFR, respectively. The differences in the training optimization of the above objective

were outlined in 3.3.2 for FULTR, and 3.4.2-3.4.3 for SPOFR. So far we have kept our notation general. Next, we will define the utility metric and fairness constraint that we will use as part of our training objective.

3.5.1 Utility Metrics

The framework proposed in this chapter can be applied to any linear utility metric U for ranking, including the Discounted Cumulative Gain (DCG), the Normalised Discounted Cumulative Gain (NDCG), Expected Reciprocal Rank (ERR), and Average Rank [71]. Since the DCG is a standard way of measuring the ranking quality of the top-k items, and naturally aligns with our evaluation objectives, we proceed with this metric:

$$DCG(r, \boldsymbol{y}_u) = \sum_{i=1}^{n} y_u^i w_{r_i}$$

where r is a permutation over n, y_u are the true relevance scores, and w is an arbitrary weight vector over ranking positions that captures the *positional discount*. These are distinct from the position bias factor v used in the calculation of group exposure, as in [49, 71, 91]. Commonly, the positional discount is calculated as:

$$w_i = \frac{1}{log_2(i+1)}$$

The positional discount reflects that items appearing lower in the ranking are penalised as the relevance score y is reduced logarithmically proportional to the position of the result. Wang et al. [132] have provided theoretical guarantees for using the logarithmic reduction factor in the DCG and NDCG.

3.5.2 Fairness Constraint

In order to produce comparable results, the end-to-end fair learning to rank models need to be trained according to the same fairness definition. Different definitions of fairness have been used across different fair in-processing learning to rank methods, as discussed in Chapter 2.

While Yadav et. al. [91] develop various amortised fairness-of-exposure constraints that can be chosen based on the needs of an application, they mainly focus on a merit-based notion of fairness founded on the relevance of a group of candidates to users. In contrast, Kotary et. al. [49] define their fairness bounds as the difference between the group and population level terms.

The definition of fairness we chose is more similar to [49]. We instead proceed with the definition of equal exposure in [70], which follows the criterion of *demographic parity*, ensuring that the average exposure from all groups is equal. We chose this definition of fairness for several reasons. First, it is not uncommon in practical recommendation settings for lower rated items to be removed from the corpus entirely before ranking. Second, because our goal in a multi-stakeholder setting is to increase the exposure of underrepresented providers, and since we can not always know *a priori* whether a provider's rating is a product of having few interactions or recommendations, relying solely on a merit-based notion of fairness may still result in an excessive bias towards the dominant group in the rankings. With this fairness criterion in mind, we can proceed to defining the fairness constraint we will use during training.

For NeuMF-SPOFR, the fairness requirement is implemented as a constraint in a linear programming model. The linear constraint takes the form of:

$$\sum_{g \neq g'} \left| \left(\frac{1}{|G_u^g|} \mathbb{1}_{G_u^g} - \frac{1}{|G_u^{g'}|} \mathbb{1}_{G_u^{g'}} \right)^\top \pi \boldsymbol{v} \right| \leq \delta$$

where, G_u^g corresponds to the set of items in a user's list that belongs to group g, $\mathbb{1}$ is the vector of all ones, and $\mathbb{1}_g$ is a vector whose values equal to 1 if the corresponding item to be ranked is in G_u^g and 0 otherwise, π corresponds to the policy matrix and v to the positional bias vector. Here we choose $v_i = 1/(1 + i)^p$ with $p = 1$. The term δ provides an upper bound on the fairness violation between each group.

For NeuMF-FULTR, we compute the group fairness coefficient at each iteration. The calculation of fairness is largely similar to the one for SPOFR, except instead of multiplying by the policy matrix term π, the disparity is calculated using only the position vector and the ranking:

$$\sum_{g \neq g'} \left| \left(\frac{1}{|G_u^g|} \mathbb{1}_{G_u^g} - \frac{1}{|G_u^{g'}|} \mathbb{1}_{G_u^{g'}} \right)^\top \boldsymbol{v} \right|$$

In order to incorporate this constraint within the gradient training of our model, we apply the log-derivative trick outlined in 3.3.2. Instead of minimising the optimization problem with respect to λ for a chosen δ, the tradeoff between utility and fairness is guided by the particular λ and then the corresponding δ is computed afterwards.

With our setting fully defined, we now turn to our experimental methodology.

Chapter 4 Methodology

In the previous section, we discussed the current state-of-the-art in fair ranking models, and formalised our methods. In this chapter, we present our proposed experiments and their expected results. Our aim is to determine which fair machine learning approaches should be used to increase provider exposures in a multi-stakeholder recommender system serving a multi-sided platform.

4.1 Dataset

For our experiments we will be using the well known Movielens 1M dataset [133] that contains 1,000,209 anonymous ratings of 3,906 movies made by 6,040 MovieLens users who joined MovieLens in 2000. This movie rating dataset has been widely used to evaluate collaborative filtering algorithms [41].

Each user in the dataset already has at least 20 ratings. Since the dataset includes explicit feedback data in the form of five star ratings, we transform it into implicit data to represent click signals, where each entry is marked as 0 or 1. We binarize relevances in the same way as [41], by assigning 1 to all the items that have an associated rating in the dataset and 0 otherwise. The implicit setting poses more difficulties for learning to rank as there is a larger presence of ties. Moreover, the 0's can either represent a missing rating due to lack of relevance or lack of exposure and thus should contribute less to the objective function, which can be mitigated through repeated sampling [88].

Since we have two learning settings in our experiment, we create two different datasets for the purpose of training and validation. In order to maintain consistency between datasets, we take the same approach to create our training, validation, and test datasets. For each user, we hold-out their latest five interactions for the test set, the previous five latest interactions for the validation set, and utilise the remaining data for training. This is a variation on leave-one-out evaluation protocol used in [41]. We use the five most recent items instead of only keeping the most recent

item for testing in order to evaluate some of the properties related to the ranking positions. We similarly segment negative items into three non-overlapping groups of user-item pairs to avoid data leakage between the different datasets.

For training with pointwise objectives, each training instance corresponds to a user-item pair. In this dataset, we have many more training and validation instances to sample from as we are not limiting the number of user-item pairs beyond the stratification used for training, validation and testing.

For the learning to rank tasks, we further transform the respective train/validation/test datasets to produce lists of 20 items per user. For training, we limit the number of relevant items to 10 per list of 20. For validation and testing, we have five relevant items per user list. This reduces the training dataset to 120,800 unique user-item pairs with 60,400 positive instances in comparison to the 939,809 positive instances in the pointwise training dataset. To increase the number of instances at training time, we randomly sample with replacement from the positive and negative items in the training set.

We further produce two testing datasets for evaluation purposes. One of the test datasets corresponds to user lists of 20 items, the other of user lists of 100 items. In each case, we retain the same five relevant items, but increase the number of negative samples. It is worth re-emphasizing that there are no overlapping user-item pairs between training, validation, and testing for any of the datasets or experiments, even for negative samples.

4.1.1 Provider Groups

The MovieLens dataset does not by itself contain information on providers. In order to obtain this information, we join the Movies Metadata dataset [134], which contains information on 45,000 movies featured in the Full MovieLens dataset, and includes the production companies. We choose production companies as a way to represent providers instead of synthetically generating providers as in [76] in order to more closely approximate a real-world distribution. Certain movies are produced by multiple production companies, in those cases we choose the most popular production company (based on number of ratings per producer in the training

dataset) to represent the provider for a given movie. We decided to follow this approach because of the impact larger movie studios have on a movie's popularity.

This leaves us with a dataset with 1,262 producers across 3,312 movies with 984,142 ratings. Initially, following the methodology in Morik et al. [60], we planned to have these production companies form the groups for which we aim to ensure fairness of exposure. However, after some initial testing, we found that trying to ensure fairness across so many producers led us to have too many lists in our training dataset that already satisfied perfect fairness of exposure (e.g. a list with 20 distinct providers is already deemed fair). As a result, we turned to the methodology used in [1]. We first calculate the number of ratings per producer using the training set and then select a cutoff threshold based on the median number of ratings per producer. For our dataset, the underrepresented group of producers, Group 0, are those with fewer or equal than 1,800 ratings and the overrepresented group of producers, Group 1, have a rating count greater than 1,800. Summary statistics for each group are presented in Table 1.

	Provider Count	**Item Count**	**User Count**	**Ratings Count**	**% of Ratings**	**Average Rating**	**Median Rating**
Group 0	1,197	1,720	6,040	346,613	35.22%	3.53	4.0
Group 1	65	1,592	6,040	637,529	64.78%	3.61	4.0

Table 1: Summary statistics per provider group.

The proportion of ratings between each group is closer than the standard long-tail cutoff. We do not split the groups 80/20 because this leads to a larger disparity in the number of items between groups and we want to maintain a more even balance between the average rating and the number of items between each group. Note that with the above split, each group exhibits similar *merit*. With this group definition, 95% of providers with 52% of items only account for 35.22% of user ratings.

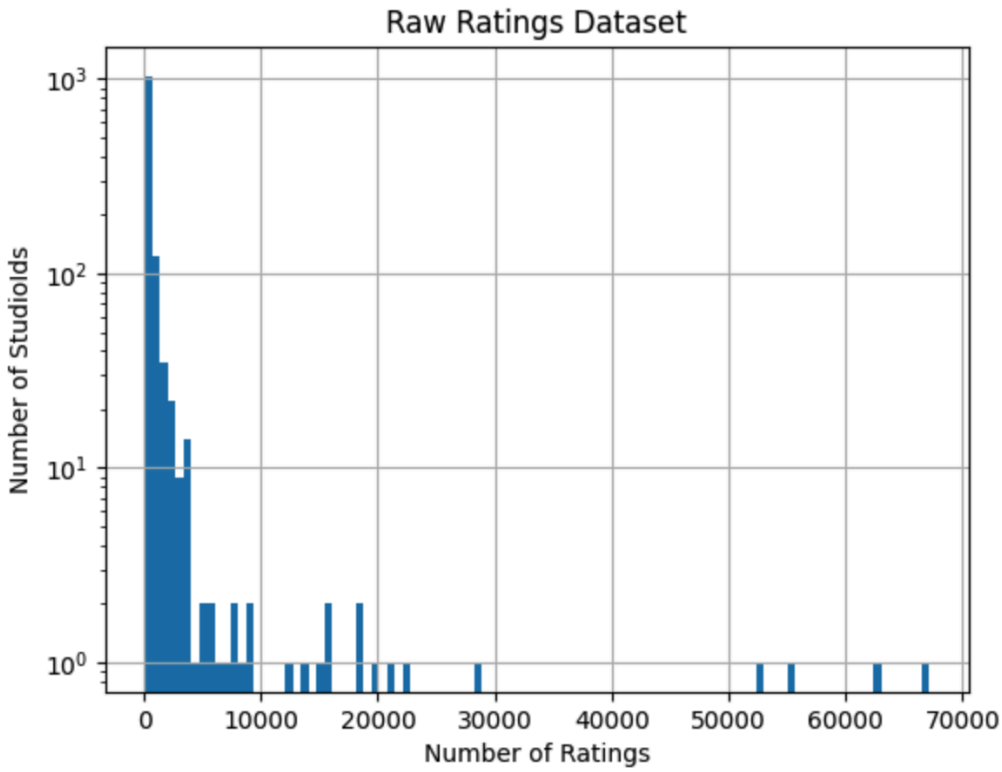


Figure 6: The count of studios bucketed by the number of ratings received in the MovieLens 1M dataset.

Figure 6 illustrates the degree of inequality between providers. We can interpret this chart as representing the inverse of the typical long-tail phenomenon for item popularity. Note that concentrated in the head of the distribution are studios that have few ratings, while the tail corresponds to popular studios with a large quantity of ratings.

In summary, our primary objective is to establish equitable recommendations across providers, ensuring that studios are recommended evenly regardless of their popularity in the existing dataset. By joining the Movies Metadata dataset with the MovieLens dataset, we incorporate production companies as a representative of providers, which allows us to more closely approximate real-world distributions of attention. We define two groups of underrepresented and overrepresented producers based on a cutoff in the count of ratings per producer in the training data. We select a lower cutoff point than the standard 80/20 split in order to maintain a more

balanced distribution of items with similar quality between each group, allowing us to measure the impact of our fairness constraint on overall attention.

4.2 Models and Hyperparameters

The hyperparameters were selected as the best-performing on average among those listed in Table 2 on the validation set.

We start by training all three Neural Collaborative Filtering (NCF) models, that is, Generalised Matrix Factorisation (GMF), Multi-Layer Perceptron MLP, and Neural Matrix Factorization (NeuMF) with 8 factors. We then train MLP with the pre-trained GMF model, and train NeuMF with both the GMF model and the pre-trained MLP model. We experiment with varying the number of factors and discuss the outcomes of increasing the number of factors in Chapter 5; however, for the purpose of the experiments we maintain 8 factors across models for ease of training and to reduce the likelihood of overfitting.

Hyperparameter	**Tested Values**
Batch size	[128, 256, 512, 1024]
Learning rate	[0.0001 ,0.0005, 0.001, 0.005]
Factors	[8, 16, 32, 64]
L2-regularisation	[0, 0.00001, 0.0001, 0.001, 0.01]
Epoch	[200] (patience of 20)
λ (FULTR)	[0.0, 0.01, 0.03, 0.1, 0.3, 1.0, 3.0, 10.0, 30.0, 100.0]
δ (SPOFR)	[0.0, 0.02, 0.04, 0.06, 0.08, 0.1, 1.0]

Table 2: Hyperparameters.

To train our pointwise model we follow the methodology in He et al. [41]. All NCF models are trained by optimising the binary cross-entropy (log loss), where we sample four negative instances per positive instance. For NCF models that are trained from scratch, we initialise model

parameters randomly according to a Gaussian distribution (with a mean of 0 and standard deviation of 0.01), optimising the model with mini-batch Adam [20].

The final hyperparameter chosen to train the GMF model were a batch size of 1024, a learning rate of 0.001 and latent factors of size 8 for pretraining. For the MLP, we trained with the following hidden layers [16, 64, 32, 16, 8]. To train the MLP, we used a batch size of 128 and a learning rate of 0.001 with a factor size of 8. For the NeuMF with pre-training, α was set to 0.5, allowing the pre-trained GMF and MLP to contribute equally to NeuMF's initialization as in [41]. We train this model with the same hyperparameters as GMF.

We train two versions of FULTR and SPOFR, one with pre-training and one without pre-training. The intuition was that pre-training would help the models learn better as it would have a better opportunity to learn the interaction function. The goal of evaluating the model both with pre-training and without is to better understand the performance of these fair learning to rank techniques when combined with latent factor models. Without special mention the underlying model architecture chosen to train all versions of FULTR and SPOFR is NeuMF. This choice was made based on the existing literature, which suggests that the NeuMF architecture has better prediction capacity compared to GMF and MLP [41, 135].

To train our FULTR model we follow the approach in Yadav et al. [91]. The expectations over ranking are approximated via S = 32 Monte-Carlo samples during training. We use the Adam optimiser with a learning rate of 0.001. We additionally subtract the average reward of the Monte-Carlo samples from the reward to act as a control variate for the variance reduction. We add the entropy of the probability distribution Softmax to the objective with a regulariser coefficient to encourage exploration and avoid convergence to suboptimal policies. We initialise the coefficient as 1.0 and reduce it by a factor of 3 each time the validation metric has stopped improving.

We train the SPOFR model with a batch size of 1024 for 200 epochs and implement patience of 10. We use the Adam optimiser with a learning rate of 0.001 and L2 regularisation of 0.01. To train our SPOFR model we replicate the implementation details in Kotary et al. [49]. For the implementation of SPOFR, we use the linear programming solver by Google OR-Tools [140]. To

increase the efficiency of training, we instantiate the solver only once at the beginning of training. For a list of length n, the number of distinct possible fairness constraints in the case of 2 groups is 2^n, which leads to unreasonable memory requirements for storing each required solver state. We follow the authors' approach to reduce the number of solvers to n solvers held in memory. We do so by exploiting the symmetry of the lists. The group identities within an item list take the form of a binary vector of length n. Once sorted, there are only n such distinct binary vectors. For an input sample and corresponding group identity vector G, let $Isort = argsort(G)$. The fairness constraints in the optimization model are then formulated based on the sorted $G' = G[Isort]$. We additionally permute the objective coefficients C by the sorting indices of G: $C' = C[Isort]$ [:]. The optimization problem is then solved using G' and C' in place of G and C. The resulting optimal policy π' then need only be reverse-permuted in its rows to restore the original orders with respect to items: $\pi = \pi'$ $[argsort(Isort)][:]$. Caching the solvers following this approach leads to substantial improvements in the training runtime.

4.3 Experimental Framework

The research questions we aim to answer with our experiments are the following:

1. Can we improve fair provider coverage while maintaining accuracy?
2. Can decision-focused learning paradigms be applied to recommender systems in order to improve balance between accuracy and other objectives?

We further breakdown these research questions into the following hypotheses:

1. End-to-end fair learning to rank techniques maintain a high performance when applied to latent factor models.
2. End-to-end learning and decision-focused learning produce better outcomes when it comes to balancing fairness and accuracy than two-stage approaches.
3. Decision-focused learning offers better control over the tradeoffs between accuracy and fairness when compared to other methods.

4. These methods are effective for multi-stakeholder scenarios, where our aim is to maximise the utility for a user, while simultaneously representing all the providers on the platform.

To evaluate the first three hypotheses, we assess our models on the test set containing 20 candidate items per user. The performance of the methods is reported on the full information test set for which all relevance labels are known.

In order to test the first hypothesis, we train the candidate models (pointwise NCF, SPOFR, FULTR) without pre-training and then with pre-training. We do this to better understand the ability of SPOFR and FULTR to learn latent factor model-based recommendation. We compare SPOFR and FULTR to the pointwise NCF on different utility measures, which we will cover in Section 4.4. The goal is to establish that FULTR and SPOFR, without fairness constraints, are competitive with conventional NCF when applied to the personalised recommendation task.

In order to test the second hypothesis, we not only rank the user lists deterministically according to the top-k scores, but also apply *Fairness of Exposure* (the constrained optimization program defined in 3.4.2) as a post-processing step. The goal is to see whether the predictions produced by the end-to-end models lead to more balanced tradeoffs between fairness and accuracy than pointwise methods, both when using a post-processing approach and when deterministically ranking the scores, and whether it has an inherently better balance between the utility and fairness objectives. Additionally, we seek to understand whether in-processing methods yield better tradeoffs between accuracy and fairness than post-processing methods.

To test the third hypothesis, we compare FULTR and SPOFR when trained with different fairness thresholds. For the baseline fair learning to rank method FULTR, this tradeoff is controlled indirectly through the constraint violation penalty term denoted λ. Higher values of λ correspond to stronger adherence to fairness. In contrast, the desired adherence to fairness can be specified explicitly in the δ-fairness constraint in the constrained optimization model of SPOFR. In order to achieve a certain level of fairness using FULTR, many values of λ must be searched until a trained model satisfying the desired fairness is found. We perform a wide hyperparameter search over λ to try and achieve similar fairness to SPOFR.

We train 17 FULTR models, each with the same configuration and vary the value of λ between [0.0, 0.01, 0.03, 0.1, 0.3, 1.0, 3.0, 10.0, 30.0, 100.0]. We then train 7 SPOFR models with the following δ values [0.0, 0.02, 0.04, 0.06, 0.08, 0.1, 1.0]. The goal is to evaluate whether SPOFR avoids the difficulties related to setting the hyperparameter λ by providing an end-to-end integration of predictions and optimization into a single machine learning pipeline.

Finally, to test our fourth hypothesis, we proceed with a different experimental design. Instead of assessing our models on the 20 candidate item test set, we turn to our 100 candidate item test set where each user list contains 5 relevant items and 95 negative samples. We still report the results on the full information test set for which all relevance labels are known. The goal of this experiment is to determine if SPOFR helps increase exposure across providers. We first simulate top-k recommendation on the dataset and produce lists of 20 items from the 100 item lists using the FairRec algorithm. FairRec, which we briefly introduced in Section 2.4.2, is an algorithmic post-processing global-wise fair ranking approach specifically designed for provider-based fairness. The FairRec algorithms are provided in Appendix B. During the experiment, we set the minimum exposure of the algorithm to 1, such that we maximise the allocation based on provider coverage. We additionally apply SPOFR post-processing to the scores and then take the top 20 items in the resulting ranking and do the same with FULTR for comparison.

4.4 Evaluation Metrics

Selecting appropriate evaluation metrics is essential as it provides a quantitative assessment of how well our models are performing. In this section, we outline the metrics we use to perform our evaluations. We start by defining our utility metrics then proceed with our fairness metrics. These metrics allow us to measure any tradeoffs between the utility of rankings and their fairness, as well as quantify the impact of these measures on provider exposure.

Utility Metrics. We evaluate the performance of a ranked list according to three utility metrics: the Hit Rate (HR), the Normalised Discounted Cumulative Gain (NDCG), and the Discounted Cumulative Gain (DCG) at the top position k.

DCG is a standard way of measuring the ranking quality of the top-k items. We define DCG@k as:

$$DCG^k = \sum_{i=1}^{k} \frac{y_u^i}{log_2(i+1)}$$

The DCG value is then normalised by the DCG of the perfect ranking, the IDCG, to obtain the NDCG metric:

$$NDCG^k = \frac{1}{IDCG^k} \cdot \sum_{i=1}^{k} \frac{y_u^i}{log_2(i+1)}$$

The HR is a binary metric that evaluates whether any of the top-k recommended items were amongst the relevant items in the test set for a given user.

$$HR^k = \max_{i=1..k} \begin{cases} 1, & r_i \in \square \\ 0, & \text{otherwise} \end{cases}$$

Fairness of Exposure. The fairness of exposure for the evaluation is calculated in the same way as during training as defined in Section 3.5.2, where we provide our motivation for this choice of criterion.

Fraction of Satisfied Producers. Here we introduce a new fairness metric to evaluate our results. We use this metric to measure how well we are meeting our objective to establish equitable recommendations across providers (producers), ensuring they are recommended evenly regardless of their popularity. A producer is satisfied if and only if its exposure is more than the minimum exposure guarantee $\bar{E} = \left\lfloor \frac{mk}{P} \right\rfloor$, where m is the number of users, k is the length of the recommended list, and P is the number of providers on the platform. The fraction of satisfied producers can be calculated as follows [63]:

$$FSP = \frac{1}{|P|} \sum_{p \in P} \mathbb{1}_{E_p \geq \bar{E}}$$

Provider Coverage. We say a provider is covered when an item in that provider's inventory belongs to at least one recommendation list:

$$C = \sum_{u \in U} \left(1 - \prod_{j \in P} 1 - \boldsymbol{x}_u p_j \right)$$

where, $\boldsymbol{x}_u$ are the items in the recommended lists and p_j is a vector of binary variables that indicate whether an item belongs to a provider or not.

This section has provided an overview of the evaluation metrics used during our experiments. Note the evaluation metrics in this thesis additionally encompass quantitative metrics that are commonly used in classification such as, precision, recall, F1, and accuracy, which we omit from the above description.

Chapter 5 Experiment Results

In this chapter we present the results of the experiments using the models presented in Chapter 3.

5.1. Can end-to-end fair learning to rank techniques work with latent factor models?

We begin our empirical evaluation by establishing that FULTR and SPOFR without fairness constraints are competitive with conventional neural collaborative filtering when applied to the collaborative ranking task. Here, we ignore fairness and entirely focus on relevancy metrics.

In Figure 7, we show the test set performance of our best NCF model (GMF without pre-training and NeuMF with pre-training) compared to FULTR with NeuMF architecture, and SPOFR with NeuMF architecture, which we call NeuMF-FULTR and NeuMF-SPOFR. In order to compare the performance of the learning to rank models with the pointwise approach, each of the models' lists are deterministically ranked based on the highest predicted value.

When it comes to ranking the 20 item lists, NeuMF-FULTR and NeuMF-SPOFR produce competitive results even without pre-training. The baseline method, GMF-Pointwise, is trained without a ranking objective and has a lower NDCG compared to the other methods. NeuMF-FULTR performs best without pre-training, this is likely due to the difference in training approach between SPOFR and FULTR. The main benefit of SPOFR is its ability to trade off fairness constraints with its utility objective. In the absence of constraints, SPOFR has a larger region that is regret-optimal for any given list, whereas FULTR's *REINFORCE* approach is well suited at maximising the empirical utility in the absence of additional constraints.

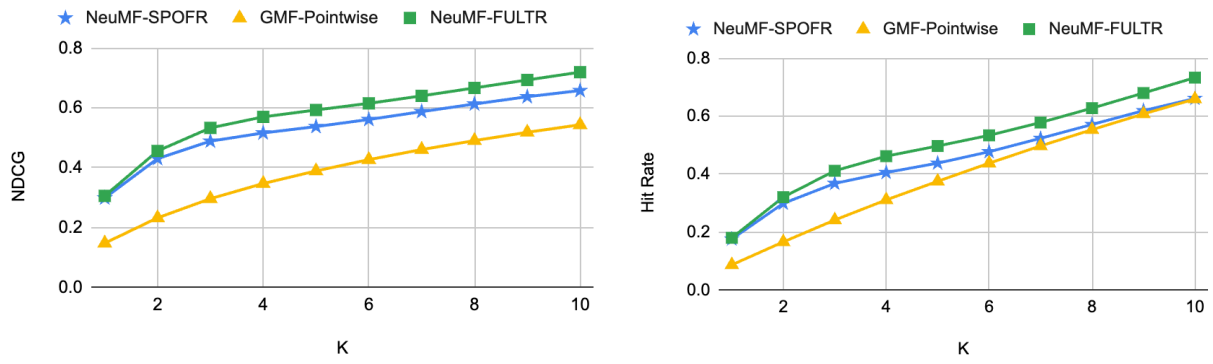


Figure 7: Evaluation of top-k item recommendation (no pre-training) on item lists of 20.

Additionally, there was not a large difference in training time for NeuMF-FULTR with pre-training or without, whereas NeuMF-SPOFR had significantly better convergence with pre-training. Pre-training significantly improved the performance of NeuMF-SPOFR and NeuMF-Pointwise, but had a negligible effect on NeuMF-FULTR.

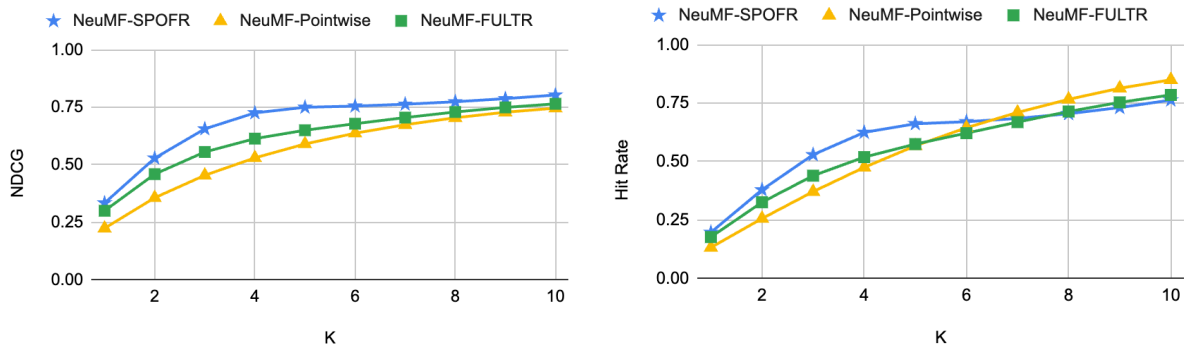


Figure 8: Evaluation of top-k item recommendation (with pre-training) on item lists of 20.

In Figure 9, we show the performance of the pre-trained models on lists larger than 20 items, unlike what was seen during training. Here we apply the models to lists of 100 items per user. While the pointwise method still has a good performance, its performance is much lower compared to NeuMF-SPOFR and NeuMF-FULTR. One possible explanation is that as learning to rank methods, the methods are better at ranking items near the top of the list. This relationship is suggested by the shape of the curves where NeuMF-SPOFR and NeuMF-FULTR plateau after the 5 relevant items are predicted.

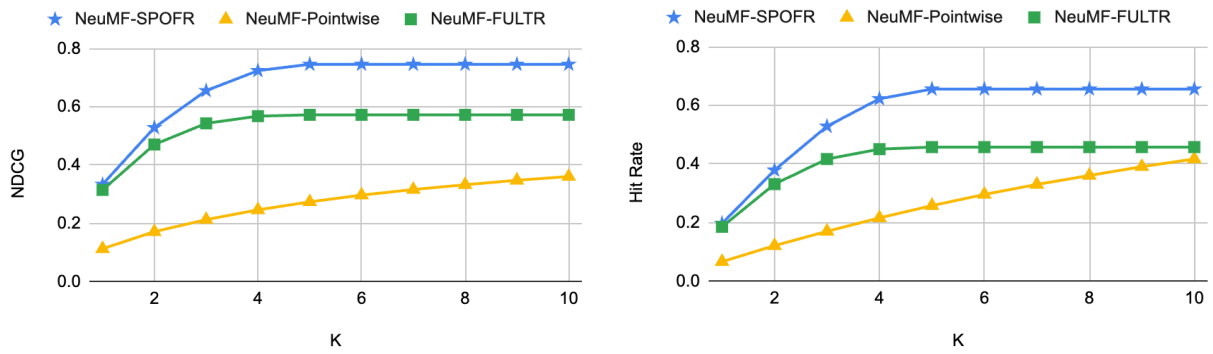


Figure 9: Evaluation of top-k item recommendation (with pre-training) on item lists of 100.

Overall, we conclude that NeuMF-FULTR and NeuMF-SPOFR are methods that achieve ranking performance on neural collaborative ranking that is competitive with pointwise NCF. The performance of these algorithms can be further improved with pre-training as in [41].

5.2. Do end-to-end learning and decision-focused learning produce better outcomes when it comes to balancing fairness and accuracy than two-stage approaches?

Next, we investigate the ability of end-to-end learning (FULTR) and decision-focused learning (SPOFR) to enforce fairness of exposure for underrepresented providers. In this section, we analyse the results on our test set of lists of 20 items per user.

We start by showing the accuracy of each method according to standard measures, summarised in Table 2. We show the cross-entropy loss (which is directly optimised by the two-stage pointwise model), the accuracy, as well as the F1 score. As expected, NeuMF-Pointwise, trained using a two-stage approach, performs best when evaluated using standard classification metrics. Additionally, NeuMF-SPOFR has the lowest performance according to standard measures out of all the trained models by some margin.

	F1		Accuracy	Cross-Entropy
	Negative	**Positive**		
NeuMF-SPOFR	70.25%	49.63%	62.60%	5.48
NeuMF-Pointwise	**82.94%**	**58.21%**	**75.77%**	**0.50**
NeuMF-FULTR	80.33%	53.15%	74.75%	2.60
Random	59.96%	33.36%	49.98%	1.00

Table 3: Accuracy of each method according to standard measures.

Table 3 shows the ranking metrics and average fairness disparity obtained when the goal is to recommend the top-k most relevant items. When deterministically ranked, NeuMF-SPOFR and NeuMF-FULTR lead to lists with lower average fairness disparity, while maintaining higher utility.

	DCG@20	NDCG@20	NDCG@5	HitRate@5	Avg Fairness Disparity
True Relevances	2.9485	1.0000	1.0000	1.0000	-0.0637
NeuMF-SPOFR	**2.6691**	**0.9053**	**0.7504**	**0.6612**	-0.0589
NeuMF-Pointwise	2.3957	0.8125	0.5908	0.5674	-0.0757
NeuMF-FULTR	2.5230	0.8557	0.6505	0.5749	-0.0602
Random	1.7615	0.5974	0.2515	0.2522	**0.000**

Table 4: DCG, NDCG, and Hit Rate of the models with scores deterministically ranked based on highest predicted value on item lists of 20.

The main test is to determine whether SPOFR performs better as a decision-focused learning approach compared to applying the same constrained optimization to the predicted values of a model trained in a two-stage approach. The intuition here is that a decision-focused method would perform better at trading off prediction errors than a two-stage approach because the model is trained using the downstream optimization task as the objective.

While we do observe this tradeoff in Figure 10, SPOFR produces higher utility lists for a given fairness threshold, the gap between approaches is relatively small. The results suggest that post-processing methods for re-ranking scores have the ability to perform similarly regardless of the training approach chosen.

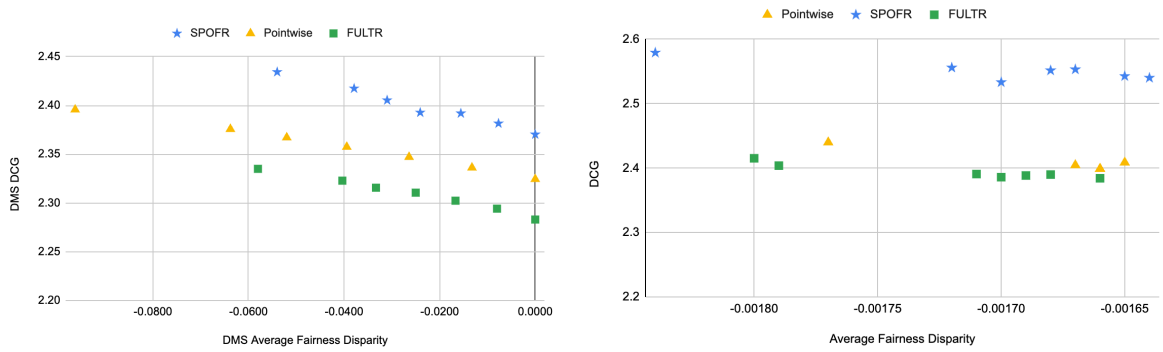


Figure 10: Evaluation of Fairness of Exposure post-processing optimization output of sampled rankings using the Birkhoff–von Neumann decomposition with metrics calculated on the doubly stochastic matrix (left) and metrics applied to the integer rankings with average fairness disparity across all users (right). Each dot is a different model for SPOFR and FULTR.

We observe that while the fairness violations due to the post-processing constrained optimization problem are lower than the fairness level guaranteed by the listed fairness parameters δ on average, when we then calculate the metrics on the rankings the average fairness disparity cannot reach zero. In comparison, FULTR was able to reach near zero disparity across queries; however, this led to a larger decrease in utility. The reason the Fairness of Exposure constrained optimization approach cannot reduce fairness of exposure disparities over the rankings to zero is likely due to a mismatch between the expected fairness disparity over multiple rankings sampled from its inferred policy and the actual fairness disparity obtained on a single sampled ranking.

	DCG	**NDCG**	**Avg Fairness Disparity**
True Relevances	2.878	0.976	-0.002
NeuMF-SPOFR	**2.371**	**0.804**	-0.002
NeuMF-Pointwise	2.321	0.787	-0.002
NeuMF-FULTR	2.283	0.776	-0.002
*FULTR**	*1.907*	*0.647*	***-0.00029***

Table 5: DCG, NDCG, and fairness disparity (averaged over all queries) of the models with Fairness of Exposure post-processing (Delta = 0). FULTR, where no Fairness of Exposure post-processing was applied, is added for comparison.*

5.3. Does decision-focused learning offer better control over the tradeoffs between accuracy and fairness when compared to other methods?

Figure 11 shows the average DCG against the average fairness disparity over the test set produced by NeuMF-SPOFR and compares with those attained by NeuMF-FULTR. Each point represents the performance of a single trained model, taken from a grid search over fairness parameters δ (for SPOFR) and λ (for FULTR). Darker colours represent more restrictive fairness parameters.

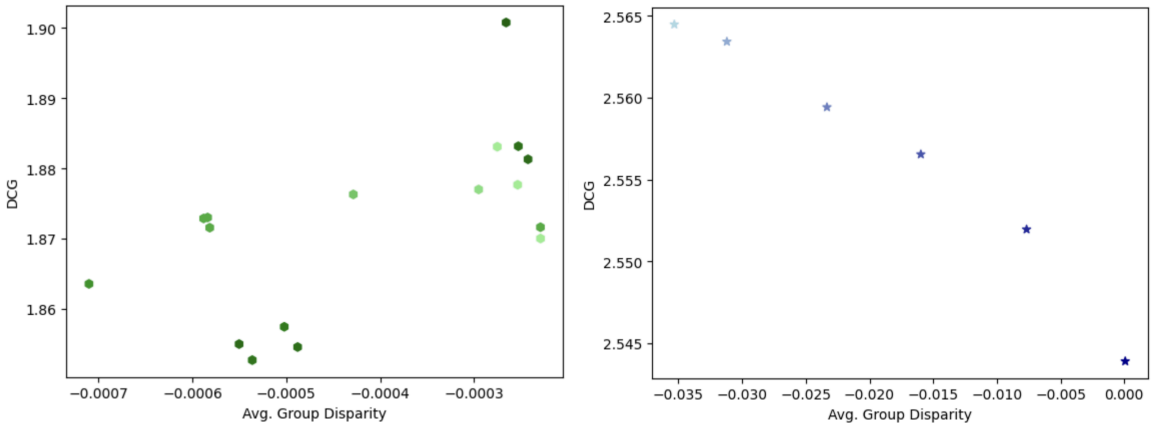


Figure 11: Fairness-utility tradeoff for NeuMF-FULTR (left) and NeuMF-SPOFR (right).

First, we observe that very low disparity could be achieved with NeuMF-FULTR, but with less control over the tradeoff between relevance and fair exposure. In contrast, as explained in the previous section, while the fairness violations produced by NeuMF-SPOFR are much lower on average than the fairness levels guaranteed by the fairness parameters, and *in expectation*, as can be seen in Figure 10, *do* reach zero, they are not as low on average compared to NeuMF-FULTR. Conversely, applying Fairness of Exposure post-processing to the NeuMF-FULTR relevances resulted in rankings with fairness disparities of -0.002 and average DCG of 2.214, a much higher DCG than was what achieved with NeuMF-FULTR standalone.

Second, the figure shows that NeuMF-SPOFR attains a substantial improvement in utility over NeuMF-FULTR, while exhibiting more consistent results across independently trained models.

In contrast, while large λ values (darker colours) should be associated with smaller fairness violations for NeuMF-FULTR compared to models trained with smaller λ values, this trend is not consistently observed. These results show the challenge in attaining a meaningful relationship between the fairness penalizer λ and the fairness violations in these state-of-the-art fair learning to rank models.

5.4. Are these methods effective for multi-stakeholder scenarios, where our aim is to maximise the utility for a user, while simultaneously representing all the providers on the platform?

To answer this question we turn to our second experiment where we ran the different models on user list sizes of 100 (with 5 relevant items and 95 negative items) and then applied different ranking and re-ranking approaches to determine the impact of in-processing methods on overall provider exposure when post-processing is applied. In particular, we compare the following post-processing methods:

1. *Top-k*: recommending the top-k relevant products,
2. *FairRec*: re-ranking algorithm based on fair resource allocation [63],
3. *SPOFR*: Fairness of Exposure is applied at inference time to rank the list,
4. *FULTR*: no additional re-ranking is applied, we sample rankings from the learned policies and select the highest probability ranking.

In the test set, Group 0 accounts for 51.9% of items, but only 36.8% of clicks. This disparity exists even though the average rating of items belonging to Group 1 is 3.63 and the average rating for Group 0 is 3.53, while the median rating for items in each group is 4.

The disparity in clicks also translates to a disparity in exposure where Group 0 has a DCG of 1.05 (35.7% of the IDCG of 2.94) and Group 1 has a DCG of 1.90. While we may think this simply represents users' choices, learning models have the potential to exacerbate differences in exposure between groups.

First, we observe in Table 6 that the NCF model trained pointwise without fairness constraints leads to an average DCG of 0.66 for Group 0, only 18% of the available exposure at the top of the list, and a DCG of 2.29 for Group 1. In contrast, even when deterministically ranked, the scores predicted by an end-to-end learning approach with fairness constraints, such as SPOFR, lead to fairer outcomes. While there is still a larger disparity than observed in the ground truth data, the DCG of Group 0 is 1.02 (31%), while the DCG of Group 1 is 1.93, these are at least no worse than existing disparities in the dataset.

	DCG@5		**NDCG@5**		**HitRate@5**		**Available Exposure**	
	Group 0	**Group 1**	**Group 0**	**Group 1**	**Group 0**	**Group 1**	**Group 0**	**Group 1**
True Relevances	1.05	1.90	0.36	0.64	1.84	3.16	36%	64%
NeuMF-SPOFR	1.02	1.93	0.34	0.66	1.83	3.17	31%	69%
NeuMF-Pointwise	0.66	2.29	0.22	0.78	1.26	3.74	18%	82%
NeuMF-FULTR	0.93	2.01	0.32	0.68	1.67	3.33	26%	72%
Random	1.55	1.40	0.53	0.47	2.61	2.39	35%	65%

Table 6: DCG, NDCG and Hit Rate and the proportion of available exposure (where click = 1) allocated to each provider group.

Second, we observe that NeuMF-SPOFR maintains high ranking utility when applied to longer lists in comparison to NeuMF-Pointwise and NeuMF-FULTR and has the lowest drop in utility when applied to the 100 item lists.

	DCG@20	**NDCG@20**	**HitRate@20**	**NDCG@5**	**HitRate@5**
NeuMF-SPOFR	2.201	0.747	0.657	0.746	0.656
NeuMF-Pointwise	1.327	0.450	0.625	0.275	0.260
NeuMF-FULTR	1.839	0.624	0.572	0.592	0.497
Random	0.355	0.120	0.200	0.05	0.05

Table 7: DCG, NDCG, and Hit Rate of the models with scores deterministically ranked based on highest predicted value on item lists of 100.

One possible explanation for the high performance of NeuMF-SPOFR is that it maintains higher recall for both the underrepresented and the overrepresented group of providers. The results appear to support the hypothesis that SPOFR is better at trading off prediction errors when compared to pointwise objectives.

	Precision		**Recall**		**F1**	
	Group 0	**Group 1**	**Group 0**	**Group 1**	**Group 0**	**Group 1**
NeuMF-SPOFR	6.63%	10.15%	67.33%	77.46%	12.07%	17.94%
NeuMF-Pointwise	12.35%	15.25%	55.84%	74.27%	20.23%	25.30%
NeuMF-FULTR	10.67%	15.15%	47.31%	63.14%	17.41%	24.44%
Random	3.49%	6.61%	50.32%	49.51%	6.52%	11.69%

Table 8: Precision, recall, and F1 scores for the positive class for each provider group (group 0 corresponds to the underrepresented group).

Finally, Table 9 compares the performance of NeuMF-SPOFR with a post-processing approach that is more targeted to the multi-stakeholder recommendation problem, FairRec. We compare the true relevances as a baseline for the tradeoff between optimal utility and producer coverage. Note that the NDCG and DCG are only computed on the 5 relevant items on a list of 20 items, so fairness to providers can be achieved without compromising utility. We have 1,262 providers in the test set and 6,040 users, meaning the minimum exposure guarantee $\overline{E}$ is equal to 95 recommendations across user lists.

First, we observe that FairRec achieves a higher fraction of satisfied producers and producer coverage for each underlying method when compared to top-k ranking; this is because FairRec tries to ensure larger exposure for producers while top-k considers only the preferences of the users. Additionally, while 100% of producers are included in at least one recommendation list when top-k is applied to the ground truth relevances, due to inherent errors in predictions, none of the models are able to cover all producers when top-k is applied to the predicted scores.

		FSP	Coverage	NDCG	DCG
True Relevances	Top-k	13.31%	100.00%	1.000	2.948
True Relevances	FairRec	46.43%	100.00%	1.000	2.948
NeuMF-SPOFR	Top-k	25.28%	66.40%	0.747	2.201
NeuMF-SPOFR		28.29%	100.00%	0.622	1.835
NeuMF-SPOFR	FairRec	36.45%	100.00%	0.503	1.483
NeuMF-Pointwise	Top-k	20.76%	36.61%	0.450	1.327
NeuMF-Pointwise	FairRec	20.68%	100.00%	0.449	1.323
NeuMF-FULTR	Top-k	22.74%	55.31%	0.592	1.839
NeuMF-FULTR		13.63%	100.00%	0.388	1.144
NeuMF-FULTR	FairRec	23.30%	100.00%	0.607	1.790

Table 9: Fraction of Satisfied Producers, producer coverage, NDCG and DCG of various models with post-processing.

Second, we observe that NeuMF-SPOFR leads to a higher fraction of satisfied producers when compared to FairRec when FairRec is applied to NeuMF-Pointwise and NeuMF-FULTR scores, all while maintaining higher utility. This indicates that listwise Fairness of Exposure can lead to better multi-stakeholder fairness even when fairness is enforced at the user-list level. Additionally, the results suggest that a post-processing approach such as FairRec can be improved by applying in-processing methods to improve fairness, even when the objectives between each are not entirely aligned.

Last, we turn to the distribution of recommendation allocation following the post-processing step FairRec. Figure 12 illustrates the long-tail phenomenon applied to producer relevance. Note, the interpretation of these charts is the reverse of that of traditional long-tail plots. Here, the x axis shows the number of recommended items and the y axis represents the number of producers per recommendation bucket. The “short head” represents providers that are rarely recommended, and the long-tail represents the providers that are frequently recommended.

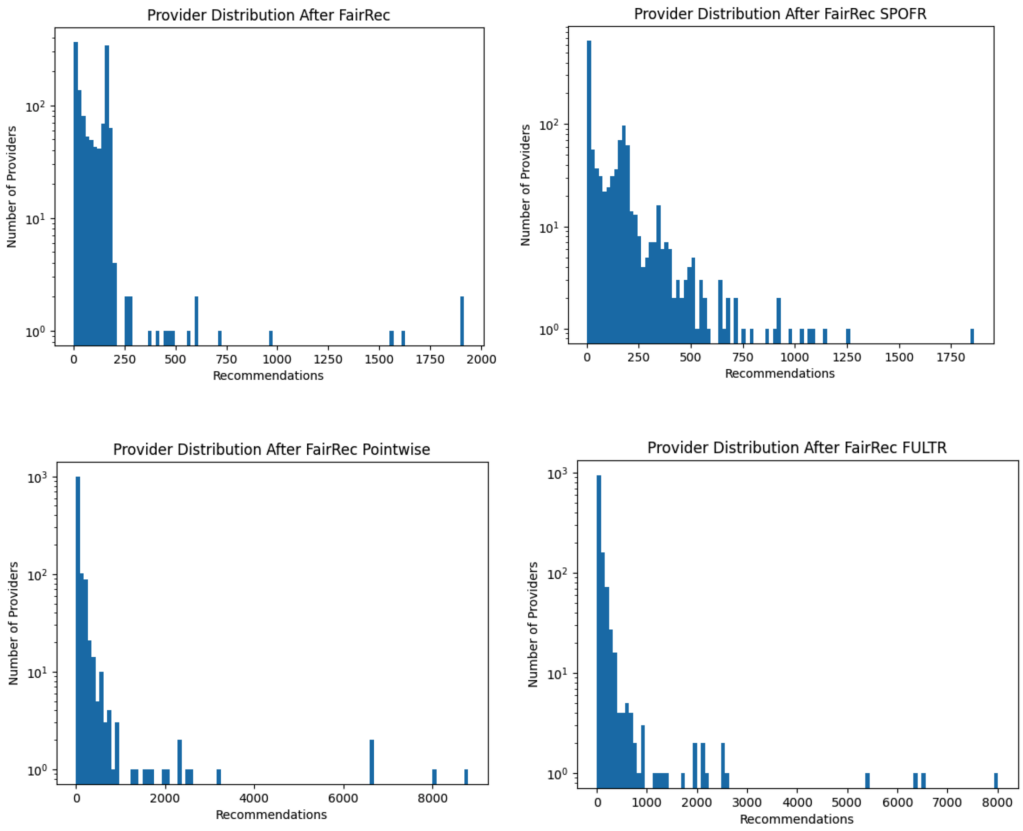


Figure 12: The long-tail of provider exposure after FairRec.

Although each of the graphs still exhibit properties of unequal allocation, we can see that FairRec helps reduce imbalances in recommendation. In the original dataset, 96% of providers have a number of user ratings below the minimum exposure guarantee. When FairRec is applied to the ground truth relevance labels, this number drops to 53.5%. When FairRec is applied to NeuMF-SPOFR, the fraction of unsatisfied producers drops to 63.5%.

Figure 13 shows the Lorenz curves for producer exposures. The cumulative fraction of total exposure is plotted against the cumulative fraction of the number of corresponding providers, ranked in increasing order of their exposure. The extent to which the curve goes below the equality mark (the dotted diagonal line) indicates the degree of inequality in the exposure distribution. We observe the Lorenz curves for top-k recommendation are far below the equal exposure marks, and are closer to the equality mark after FairRec.

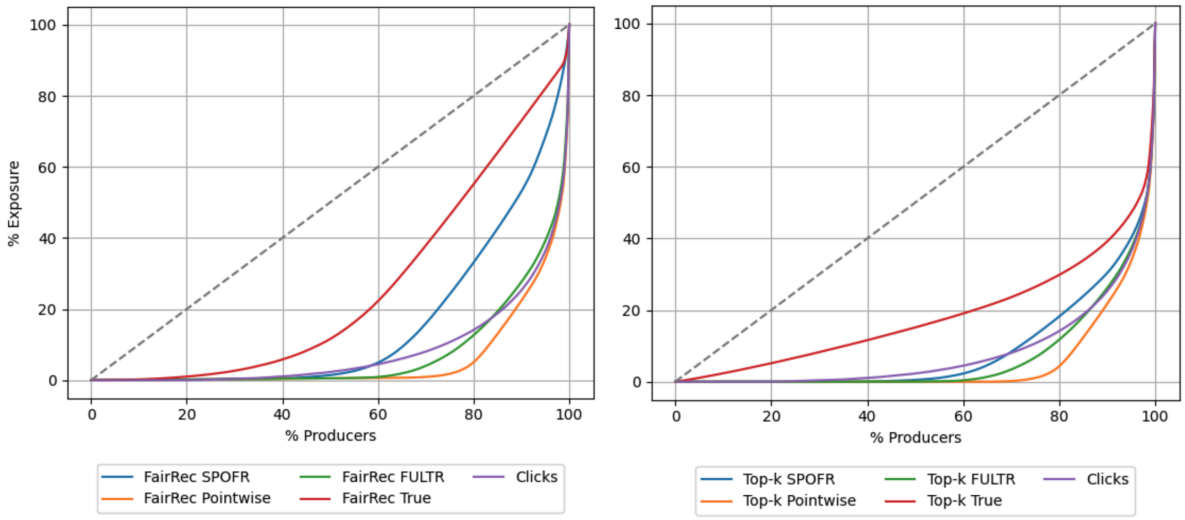


Figure 13: Lorenz curves plotting cumulative fraction of total exposure against the cumulative fraction of the number of providers. FairRec (left) and Top-k ranking (right).

Note, there is virtually no difference in the Lorenz curves between FairRec and top-k re-ranking when applied to the pointwise scores. FairRec first attempts to provide a minimum guarantee on the exposure of the producers. Then it iteratively assigns to the selected user the remaining unallocated items by prioritising items that are most relevant to the user. Because the NeuMF-Pointwise model has significantly better recall for the dominant group, items belonging to this group are prioritised in the second run, which leads to a more uneven allocation of exposure across providers. Based on previous observations, this helps reinforce the notion that training without sufficient bias mitigation can lead to inequalities in group exposure, which could worsen over time.

Overall, we conclude that NeuMF-SPOFR, with and without FairRec post-processing, can greatly help improve fair exposure allocation amongst providers, and that decision-focused learning offers an inherently better tradeoff between fairness to providers and utility to the end user.

Chapter 6 Conclusion

In our work, we aimed to determine whether decision-focused learning could be successfully applied to the domain of fair recommendation in order to improve the balance between accuracy and fairness. Additionally, we sought to establish the effectiveness of balancing accuracy and fairness in multi-stakeholder scenarios, where our aim is to maximise the utility for a user, while simultaneously representing all the providers on the platform.

To determine whether decision-focused learning could be applied to the domain of fair recommendation, we successfully combined an existing decision-focused technique for fairness-constrained learning to rank, SPOFR [49], and applied it to the setting of collaborative ranking using neural collaborative filtering. Specifically, we combined SPOFR with a Neural Matrix Factorisation (NeuMF) model [41] to create NeuMF-SPOFR. We further combined NeuMF with another state-of-the-art in-processing fair learning to rank technique, FULTR [91], to create NeuMF-FULTR.

Our empirical results establish that NeuMF-FULTR and NeuMF-SPOFR are competitive with conventional neural collaborative filtering when applied to the collaborative ranking task, and indicate that the performance of these algorithms can be further improved with pre-training as in [41]. To the best of our knowledge, our work is the first to merge neural collaborative filtering, a prevalent deep learning technique for personalised recommendation, with in-processing fair learning to rank methods.

One of the key objectives of our experiments was to assess the extent to which decision-focused learning enables finer control over the tradeoffs between accuracy and fairness, in comparison to regularisation-based in-processing methods. To address this question, we conducted a comparison between FULTR and SPOFR, and our findings corroborated the results reported in [49]. Our results highlight the difficulty in establishing a meaningful relationship between the fairness penalizer λ and fairness violations in state-of-the-art fair learning to rank models. This underscores the significance of decision-focused learning in providing control over the tradeoffs between accuracy and fairness.

We additionally investigated the ability of end-to-end learning (FULTR) and decision-focused learning (SPOFR) to enforce fairness of exposure for underrepresented providers. The intuition was that decision-focused learning would perform better at trading off prediction errors than a two-stage approach because the model would be trained using the downstream optimization task as the objective. While we did observe this tradeoff to some extent, the results suggested that post-processing methods for re-ranking scores can help improve fairness of exposure, even with two-stage training.

In our final experimental setting, decision-focused learning was able to strike a superior tradeoff between fairness to providers and utility to the end user. Notably, our results showcased that enhancing fairness through in-processing techniques can lead to a more equitable allocation of exposure, even when fairness is enforced at the user-list level during training. This highlights the inherent strength of decision-focused learning in achieving fairness objectives while simultaneously optimising utility, thereby ensuring a fairer and more satisfying recommendation experience for both providers and users.

In summary, our findings indicate that NeuMF-SPOFR, both with and without post-processing, exhibits remarkable potential in enhancing fair exposure allocation among providers. Moreover, we observe that decision-focused learning inherently offers a finer control over the balance between fairness to providers and utility to the end user, making this method attractive for multi-stakeholder recommendation in multi-sided platforms.

6.1 Future Works

While this study has made progress in exploring the application of decision-focused learning for fair multi-stakeholder recommendation, there are several avenues for future research and development in this field. The following are potential directions for future works:

Exploring Alternative Fairness Metrics. In this study, we focused on balancing fairness and accuracy in the context of multi-stakeholder recommendation, and chose a criterion of *demographic parity* in order to ensure that the average exposure from all groups is equal. However, there are various fairness metrics and definitions that can be explored, including more

merit-based notions of fairness [78]. Future research can investigate the application of decision-focused learning techniques to optimise these alternative fairness metrics.

Incorporating User Feedback. Our fairness framework focused on *treatment-based fairness* [81], which only considers whether the treatments of the recommender system are fair or not, such as the allocated exposure to different providers. A future avenue for this work would be to investigate *impact-based fairness* by incorporating user feedback into the framework. Initially, our intention was to replicate the *click simulation* proposed by Yadav et. al. [91] to reproduce biases inherent to implicit feedback settings [45]. Although we successfully generated the click data and conducted some training, we were unable to allocate sufficient time to report on the obtained results.

Considering Dynamic Environments. Additionally, given that in practical settings user preferences are dynamic and constantly evolving [60], future research can focus on developing decision-focused learning techniques that can adapt and learn in real-time, considering the changing dynamics of the multi-stakeholder environment. Another possible avenue, given the computational overhead of decision-focused learning techniques, would be to simulate the longitudinal effects of fairness interventions on multi-stakeholder environments and prodivers' long term success.

Construction of Solution Cache. The primary drawback of decision-focused techniques, such as SPOFR, is that they require solving a linear programming problem for each training instance, and additionally at inference time. Not only does this limit their ability to rank lists of arbitrary size, as runtime increases quadratically with the size of the list to be ranked, it also restricts their feasibility for real-time deployment in both training and inference scenarios. While our work has found that solution caching and pre-training greatly alleviates training time, it remains to be seen whether hot-starting schemes [101] or similar caching-based approaches can be leveraged for model serving.

Real-world Deployment. Finally, while this study provides promising results, further research is needed to evaluate the proposed decision-focused learning techniques in real-world multi-stakeholder platforms.

Bibliography

[1] Abdollahpouri, H., Burke, R., & Mobasher, B. (2017). Controlling Popularity Bias in Learning-To-Rank Recommendation. In *Proceedings of the 11th ACM Conference on Recommender Systems.* (pp. 42-46).

[2] Abdollahpouri H, Adomavicius G, Burke R, Guy I, Jannach D, Kamishima T, Krasnodebski J, Pizzato L. (2020). Multistakeholder recommendation: Survey and research directions. *User Modeling and User-Adapted Interaction*, *30*. (pp. 127-158).

[3] Adams, R. P., & Zemel, R. S. (2011). Ranking via sinkhorn propagation. *arXiv preprint arXiv:1106.1925*.

[4] Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., & Wallach, H. (2018). A Reductions Approach to Fair Classification. In *Proceedings of the 35th International Conference on Machine Learning, PMLR.* (pp. 60-69).

[5] Aggarwal, C. C. (2016). *Recommender systems (Vol. 1)*. Springer International Publishing.

[6] Bartels, N., & Schmitt, A. (2022). Developing network effects for digital platforms in two-sided markets–The NfX construction guide. In *Digital Business, 2(2)*. (pp. 100044).

[7] Beutel, A., Chen, J., Doshi, T., Qian, H., Wei, L., Wu, Y., Heldt, L., Zhao, Z., Hong, L., Chi, E.H. & Goodrow, C. (2019). Fairness in Recommendation Ranking through Pairwise Comparisons. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* (pp. 2212-2220).

[8] Bi, J. W., Liu, Y., & Fan, Z. P. (2020). A deep neural networks based recommendation algorithm using user and item basic data. *International journal of machine learning and cybernetics*, *11*. (pp. 763-777).

[9] Biega, A. J., Gummadi, K. P., & Weikum, G. (2018). Equity of Attention: Amortizing Individual Fairness in Rankings. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*. (pp. 405-414).

[10] Birkhoff, G. (1946). Three observations on linear algebra. *Univ. Nac. Tacuman, Rev. Ser. A, 5.* (pp. 147-151).

[11] Blondel, M., Teboul, O., Berthet, Q., & Djolonga, J. (2020). Fast Differentiable Sorting and Ranking. In *Proceedings of the 37th International Conference on Machine Learning.* (pp. 950-959).

[12] Bobadilla, J., Ortega, F., Hernando, A., & Bernal, J. (2012). A collaborative filtering approach to mitigate the new user cold start problem. *Knowledge-based systems*, *26*. (pp. 225-238).

[13] Bower, A., Lum, K., Lazovich, T., Yee, K., & Belli, L. (2022). Random Isn't Always Fair: Candidate Set Imbalance and Exposure Inequality in Recommender Systems. *arXiv preprint arXiv:2209.05000.*

[14] Bruch, S., Han, S., Bendersky, M., & Najork, M. (2020). A Stochastic Treatment of Learning to Rank Scoring Functions. In *Proceedings of the 13th International Conference on Web Search And Data Mining.* (pp. 61-69).

[15] Burke, R., Sonboli, N., & Ordonez-Gauger, A. (2018). Balanced Neighborhoods for Multi-sided Fairness in Recommendation. In *Proceedings of the 1st Conference on Fairness, Accountability and Transparency, PMLR*. (pp. 202-214).

[16] Burke, R.: Multisided Fairness for Recommendation. (2017). In *4th Workshop on Fairness, Accountability, and Transparency in Machine Learning (FAT/ML 2017)*.

[17] Burke, R. D., Abdollahpouri, H., Mobasher, B., & Gupta, T. (2016). Towards multi-stakeholder utility evaluation of recommender systems. In *Workshop on Surprise, Opposition, and Obstruction in Adaptive and Personalized Systems, SOAP 2016.* (pp. 750).

[18] Cao, Z., Qin, T., Liu, T. Y., Tsai, M. F., & Li, H. (2007). Learning to Rank: From Pairwise Approach to Listwise Approach. In *Proceedings of the 24th International Conference on Machine Learning.* (pp. 129-136).

[19] Caton, S., & Haas, C. (2023). Fairness in machine learning: A survey. *ACM Computing Surveys*. Advance online publication. https://doi.org/10.1145/3616865

[20] Celis, L. E., Straszak, D., & Vishnoi, N. K. (2018). Ranking with fairness constraints. In *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP '18)*.

[21] Chan, D., Voortman, F., & Rogers, S. (2019). Deloitte: The Rise of the Platform Economy. Retrieved from https://www2.deloitte.com/content/dam/Deloitte/nl/Documents/humancapital/deloitte-nl-hc-the-rise-of-the-platform-economy-report.pdf

[22] Cossock, D., & Zhang, T. (2006). Subset ranking using regression. In *Learning Theory: 19th Annual Conference on Learning Theory, COLT 2006, Proceedings 19.* (pp. 605-619).

[23] Covington, P., Adams, J., & Sargin, E. (2016). Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM conference on recommender systems.* (pp. 191-198).

[24] Deldjoo, Y., Jannach, D., Bellogin, A., Difonzo, A., & Zanzonelli, D. (2023). Fairness in Recommender Systems: Research Landscape and Future Directions. *User Modeling and User-Adapted Interaction*. (pp. 1-50).

[25] Diaz, F., Mitra, B., Ekstrand, M. D., Biega, A. J., & Carterette, B. (2020). Evaluating stochastic rankings with expected exposure. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management.* (pp. 275-284).

[26] Do, V., Corbett-Davies, S., Atif, J., & Usunier, N. (2021). Two-sided fairness in rankings via Lorenz dominance. *Advances in Neural Information Processing Systems, 34*. (pp. 8596-8608).

[27] El Balghiti, O., Elmachtoub, A. N., Grigas, P., & Tewari, A. (2019). Generalization Bounds in the Predict-Then-Optimize Framework. *Advances in neural information processing systems, 32*. (pp. 14412–14421)

[28] Elmachtoub, A. N., & Grigas, P. (2022). Smart "Predict, then Optimize". *Management Science, 68(1)*. (pp. 9-26).

[29] End to End Learning and Optimization. *AI Institute for Advances in Optimization*. Retrieved from https://www.ai4opt.org/end-end-learning-and-optimization

[30] Evans, D. S., Schmalensee, R., Noel, M. D., Chang, H. H., & Garcia-Swartz, D. D. (2011). Platform Economics: Essays on Multi-Sided Businesses. In: D. S. Evans (Ed.), Competition Policy International.

[31] Evans, D.S., Schmalensee, R. (2017). Multi-sided Platforms. In: The New Palgrave Dictionary of Economics. Palgrave Macmillan, London. https://doi.org/10.1057/978-1-349-95121-5_3069-1

[32] García-Soriano, D., & Bonchi, F. (2021). Maxmin-Fair Ranking: Individual Fairness under Group-Fairness Constraints. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining.* (pp. 436-446).

[33] Ge, Y., Liu, S., Gao, R., Xian, Y., Li, Y., Zhao, X., Pei, C., Sun, F., Ge, J., Ou, W., & Zhang, Y. (2021). Towards Long-term Fairness in Recommendation. In *Proceedings of the 14th ACM International Conference on Web Search & Data Mining*. (pp. 445-453).

[34] Geyik, S. C., Ambler, S., & Kenthapadi, K. (2019). Fairness-Aware Ranking in Search & Recommendation Systems with Application to LinkedIn Talent Search. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. (pp. 2221–2231).

[35] Gomez-Uribe, C. A., & Hunt, N. (2015). The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Transactions on Management Information Systems (TMIS)*, *6*(4). (pp. 1-19).

[36] Gorantla, S., Bhansali, E., Deshpande, A., & Louis, A. (2023). Optimizing Group-Fair Plackett-Luce Ranking Models for Relevance and Ex-Post Fairness. *arXiv preprint arXiv:2308.13242*.

[37] Gorantla, S., Deshpande, A., & Louis, A. (2022). Sampling Ex-Post Group-Fair Rankings. *arXiv e-prints, arXiv-2203.00887*

[38] Gorantla, S., Mehrotra, A., Deshpande, A., & Louis, A. (2023). Sampling Individually-Fair Rankings that are Always Group Fair. In *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society (AIES '23)*. (pp. 205–216).

[39] Govindarajan, V., & Venkatraman, N. V. (2022). The Next Great Digital Advantage. *Harvard Business Review, 100(5-6)*. (pp. 56-63).

[40] Guo, G. (2012). Resolving Data Sparsity and Cold Start in Recommender Systems. In *User Modeling, Adaptation, and Personalization: 20th International Conference, UMAP 2012, Proceedings 20* (pp. 361-364).

[41] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017, April). Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web*. (pp. 173-182).

[42] Heuss, M., Sarvi, F., & de Rijke, M. (2022). Fairness of Exposure in Light of Incomplete Exposure Estimation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval.* (pp. 759-769).

[43] Hoyos-Idrobo, A. (2020). Approximate Birkhoff-von-Neumann Decomposition: a Differentiable Approach.

[44] Isinkaye, F. O., Folajimi, Y. O., & Ojokoh, B. A. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian informatics journal*, *16*(3). (pp. 261-273).

[45] Joachims, T., Swaminathan, A., & Schnabel, T. (2017). Unbiased Learning-to-Rank with Biased Feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining.* (pp. 781-789).

[46] Jones, A. Log-Derivative Trick. *Andy Jones*. Retrieved from https://andrewcharlesjones.github.io/journal/log-derivative.html

[47] Klimashevskaia, A., Jannach, D., Elahi, M., & Trattner, C. (2023). A Survey on Popularity Bias in Recommender Systems. *arXiv preprint arXiv:2308.01118*.

[48] Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8). (pp. 30-37).

[49] Kotary, J., Fioretto, F., Van Hentenryck, P., & Zhu, Z. (2022). End-To-End Learning For Fair Ranking Systems. In *Proceedings of the ACM Web Conference 2022.* (pp. 3520-3530).

[50] Li, F., Qu, H., Fu, M., Zhang, L., Zhang, F., Chen, W., Sun, R., & Zhang, H. (2022). Neural Reranking-Based Collaborative Filtering by Leveraging Listwise Relative Ranking Information. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *53*(2). (pp. 882-896).

[51] Li, H. (2022). *Learning to rank for information retrieval and natural language processing*. Springer Nature.

[52] Li, Y., Chen, H., Xu, S., Ge, Y., Tan, J., Liu, S., & Zhang, Y. (2023). Fairness in Recommendation: Foundations, Methods, and Applications. *ACM Transactions on Intelligent Systems and Technology*, *14*(5). (pp. 1-48).

[53] Liu, T. Y. (2009). Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, *3*(3). (pp. 225-331).

[54] Luce, R. D. (2012). Individual Choice Behavior: A Theoretical Analysis. *Courier Corporation.*

[55] Mahapatra, D., Dong, C., & Momma, M. (2023). Querywise Fair Learning to Rank through Multi-Objective Optimization. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining.* (pp. 1653–1664).

[56] Mandi, J., Kotary, J., Berden, S., Mulamba, M., Bucarey, V., Guns, T., & Fioretto, F. (2023). Decision-Focused Learning: Foundations, State of the Art, Benchmark and Future Opportunities. *arXiv preprint arXiv:2307.13565.*

[57] Martins, A., & Astudillo, R. (2016). From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning* . (pp. 1614-1623).

[58] Mehrotra, A., & Vishnoi, N. (2022). Fair Ranking with Noisy Protected Attributes. In *Advances in Neural Information Processing Systems 35*. (pp. 31711-31725)

[59] Mladenov, M., Creager, E., Ben-Porat, O., Swersky, K., Zemel, R., & Boutilier, C. (2020). Optimizing Long-term Social Welfare in Recommender Systems: A Constrained Matching Approach. In *Proceedings of the 37th International Conference on Machine Learning, PMLR.* (pp. 6987-6998).

[60] Morik, M., Singh, A., Hong, J., & Joachims, T. (2020). Controlling Fairness and Bias in Dynamic Learning-to-Rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* (pp. 429-438).

[61] Narasimhan, H., Cotter, A., Gupta, M., & Wang, S. (2020). Pairwise Fairness for Ranking and Regression. In *Proceedings of the AAAI Conference on Artificial Intelligence, 34(4)*. (pp. 5248-5255).

[62] Oosterhuis, H. (2021). Computationally Efficient Optimization of Plackett-Luce Ranking Models for Relevance and Fairness. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval.* (pp. 1023-1032).

[63] Patro, G. K., Biswas, A., Ganguly, N., Gummadi, K. P., & Chakraborty, A. (2020). FairRec: Two-Sided Fairness for Personalized Recommendations in Two-Sided Platforms. In *Proceedings of the Web Conference 2020.* (pp. 1194-1204).

[64] Plackett, R. L. (1975). The Analysis of Permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics, 24(2)*. (pp. 193-202).

[65] Rahimi, R., Montazeralghaem, A., & Allan, J. (2019). Listwise Neural Ranking Models. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval.* (pp. 101-104).

[66] Ricci, F., Rokach, L., & Shapira, B. (2010). Introduction to recommender systems handbook. In *Recommender systems handbook* (pp. 1-35).

[67] Rochet, J.-C., & Tirole, J. (2003). Platform Competition in Two-sided Markets. *Journal of the European Economic Association*, *1*(4). (pp. 990–1029).

[68] Santucci, V., & Ceberio, J. (2023). Doubly Stochastic Matrix Models for Estimation of Distribution Algorithms. *arXiv preprint arXiv:2304.02458.*

[69] Shi, Y., Larson, M., & Hanjalic, A. (2010). List-wise learning to rank with matrix factorization for collaborative filtering. In *Proceedings of the fourth ACM International Conference on Recommender Systems.* (pp. 269-272).

[70] Singh, A., & Joachims, T. (2018). Fairness of Exposure in Rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. (pp. 2219-2228).

[71] Singh, A., & Joachims, T. (2019). Policy Learning for Fairness in Ranking. *Advances in Neural Information Processing Systems 32*. (pp. 5427–5437).

[72] Singh, A., Kempe, D., & Joachims, T. (2021). Fairness in Ranking under Uncertainty. In *Advances in Neural Information Processing Systems 34*. (pp. 11896-11908).

[73] Solsman, Joan E. (2018, January 10). YouTube's AI is the puppet master over most of what you watch. *CNET*. Retrieved from https://www.cnet.com/tech/services-and-software/youtube-ces-2018-neal-mohan/

[74] Sonboli, N., Burke, R., Ekstrand, M., & Mehrotra, R. (2022). The Multisided Complexity of Fairness in Recommender Systems. *AI magazine, 43(2).* (pp. 164-176).

[75] Sühr, T., Biega, A. J., Zehlike, M., Gummadi, K. P., & Chakraborty, A. (2019). Two-Sided Fairness for Repeated Matchings in Two-Sided Markets: A Case Study of a Ride-Hailing Platform. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* (pp. 3082-3092).

[76] Sürer, Ö., Burke, R., & Malthouse, E. C. (2018). Multistakeholder Recommendation with Provider Constraints. In *Proceedings of the 12th ACM Conference on Recommender Systems*. (pp. 54-62).

[77] Trischler, M. F. G., Meier, P., & Trabucchi, D. (2021). Digital platform tactics: How to implement platform strategy over time. In *Journal of Business Models, 9(1)*. (pp. 67-76).

[78] Usunier, N., Do, V., & Dohmatob, E. (2022). Fast Online Ranking With Fairness Of Exposure. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency.* (pp. 2157-2167).

[79] Vorotilov V., & Shugaepov, I. (2023, August 9). Scaling the Instagram Explore recommendations system. *Engineering at Meta.* Retrieved from

https://engineering.fb.com/2023/08/09/ml-applications/scaling-instagram-explore-recommendations-system/

[80] Wang, L., Lin, J., & Metzler, D. (2011). A cascade ranking model for efficient ranked retrieval. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. (pp. 105-114).

[81] Wang, Y., Ma, W., Zhang, M., Liu, Y., & Ma, S. (2023). A Survey on the Fairness of Recommender Systems. *ACM Transactions on Information Systems*, *41*(3). (pp. 1-43).

[82] Wang, Y., Tao, L., & Zhang, X. X. (2023). Recommending for a multi-sided marketplace: A multi-objective hierarchical approach. *Available at SSRN 4602954*.

[83] What is end-to-end learning in AI? *TED AI 2023*. Retrieved from https://www.ai-event.ted.com/glossary/end-to-end-learning

[84] Wilder, B., Dilkina, B., & Tambe, M. (2019). Melding the Data-Decisions Pipeline: Decision-Focused Learning for Combinatorial Optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence, 33(1)*. (pp. 1658-1665).

[85] Williams, R. J. (1992). Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine learning, 8*. (pp. 229-256).

[86] Wu, H., Ma, C., Mitra, B., Diaz, F., & Liu, X. (2022). A Multi-Objective Optimization Framework for Multi-Stakeholder Fairness-Aware Recommendation. *ACM Transactions on Information Systems 41(2)*. (pp. 1-29).

[87] Wu, H., Mitra, B., Ma, C., Diaz, F., & Liu, X. (2022). Joint Multisided Exposure Fairness For Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval.* (pp. 703-714).

[88] Wu, L., Hsieh, C. J., & Sharpnack, J. (2018). SQL-Rank: A Listwise Approach to Collaborative Ranking. In *International Conference on Machine Learning, PMLR*. (pp. 5315-5324).

[89] Wu, L. (2020). Advances in Collaborative Filtering and Ranking. *arXiv preprint arXiv:2002.12312(2020)*.

[90] Wu, Y., Cao, J., Xu, G., & Tan, Y. (2021). TFROM: A Two-sided Fairness-Aware Recommendation Model for Both Customers and Providers. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. (pp. 1013-1022).

[91] Yadav, H., Du, Z., & Joachims, T. (2021). Policy-Gradient Training of Fair and Unbiased Ranking Functions. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval.* (pp. 1044-1053).

[92] Yao, S., & Huang, B. (2017). Beyond Parity: Fairness Objectives for Collaborative Filtering. *Advances in Neural Information Processing Systems 30.* (pp. 2921–2930).

[93] Zehlike, M., & Castillo, C. (2020). Reducing Disparate Exposure in Ranking: A Learning To Rank Approach. In *Proceedings of the Web Conference 2020.* (pp. 2849-2855).

[94] Zehlike, M., Bonchi, F., Castillo, C., Hajian, S., Megahed, M., & Baeza-Yates, R. (2017). FA*IR: A Fair Top-k Ranking Algorithm. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17).* (pp. 1569-1578).

[95] Zehlike, M., Yang, K., & Stoyanovich, J. (2022). Fairness in Ranking: A Survey. *ACM Computing Surveys, 55(6)*. (pp. 1-41).

[96] Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2023). Dive into Deep Learning. *Cambridge University Press.*

[97] Zhao, Y., Von Delft, S., Morgan-Thomas, A., & Buck, T. (2020). The evolution of platform business models: Exploring competitive battles in the world of platforms. *Long Range Planning*, *53*(4). (pp. 101892).

[98] Zou, H., Zhu, J., & Hastie, T. (2008). New Multicategory Boosting Algorithms Based on Multicategory Fisher-Consistent Losses. *The Annals of Applied Statistics, 2(4)*. (pp. 1290).

[99] Singh, A. (2021). Fairness of Exposure for Ranking Systems. Ph.D. Dissertation. Cornell University.

[100] Ekstrand, M. D., Das, A., Burke, R., & Diaz, F. (2022). Fairness in information access systems. *Foundations and Trends® in Information Retrieval, 16(1-2)*. (pp. 1-177).

[101] Mandi, J., Stuckey, P. J., & Guns, T. (2020). Smart predict-and-optimize for hard combinatorial optimization problems. In P*roceedings of the AAAI Conference on Artificial Intelligence 34(02).* (pp. 1603-1610).

[102] Wan, M., Zha, D., Liu, N., & Zou, N. (2023). In-processing modeling techniques for machine learning fairness: A survey. *ACM Transactions on Knowledge Discovery from Data, 17(3).* (pp. 1-27).

[103] Agarwal, A., Zaitsev, I., Wang, X., Li, C., Najork, M., & Joachims, T. (2019). Estimating position bias without intrusive interventions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining.* (pp. 474-482).

[104] Chakraborty, A., Hannak, A., Biega, A. J., & Gummadi, K. (2017). Fair sharing for sharing economy platforms. In *Fairness, Accountability and Transparency in Recommender Systems-Workshop on Responsible Recommendation.*

[105] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., & Zemel, R. (2012). Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference.* (pp. 214-226).

[106] Narasimhan, H. (2018). Learning with complex loss functions and constraints. In *International Conference on Artificial Intelligence and Statistics.* (pp. 1646-1654).

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

[108] Rezaei, A., Fathony, R., Memarrast, O., & Ziebart, B. (2020, April). Fairness for robust log loss classification. In *Proceedings of the AAAI Conference on Artificial Intelligence 34(04)*. (pp. 5511-5518).

[109] Lerche, L. (2016). Using implicit feedback for recommender systems: characteristics, applications, and challenges.

[110] Hardt, M., Price, E., & Srebro, N. (2016). Equality of opportunity in supervised learning. *Advances in neural information processing systems, 29.*

[111] Joachims, T., Granka, L., Pan, B., Hembrooke, H., & Gay, G. (2017). Accurately interpreting clickthrough data as implicit feedback. In *ACM SIGIR Forum 51(01).* (pp. 4-11).

[112] Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* (pp. 133–142).

[113] Joachims, T. (2006). Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* (pp. 217-226).

[114] Shams, B., & Haratizadeh, S. (2017). Graph-based collaborative ranking. *Expert Systems with Applications, 67*. (pp. 59-70)

[115] Zliobaite, Indre. (2015). On the relation between accuracy and fairness in binary classification. In *Proceedings Of The 2nd Workshop On Fairness, Accountability, And Transparency In Machine Learning.*

[116] Craswell, N., Zoeter, O., Taylor, M., & Ramsey, B. (2008). An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining.* (pp. 87-94).

[117] Taylor, M., Guiver, J., Robertson, S., & Minka, T. (2008). Softrank: optimizing non-smooth rank metrics. In *Proceedings of the 2008 International Conference on Web Search and Data Mining.* (pp. 77-86).

[118] Burges, C., Ragno, R., & Le, Q. (2006). Learning to rank with nonsmooth cost functions. *Advances in neural information processing systems*, *19*.

[119] Burges, C. J. (2010). From ranknet to lambdarank to lambdamart: An overview. In *Microsoft Research Technical Report MSR-T R-2010–82.*

[120] Marcuzzo, M., Zangari, A., Albarelli, A., & Gasparetto, A. (2022). Recommendation systems: an insight into current development and future research challenges. *IEEE Access, 10.* (pp. 86578-86623).

[121] Chen, M., & Zhou, X. (2020). DeepRank: Learning to rank with neural networks for recommendation. *Knowledge-Based Systems, 209*. (pp. 106478).

[122] Amos, B., & Kolter, J. Z. (2017, July). Optnet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70 (ICML'17)*. (pp. 136-145).

[123] Kuhn, H. W., & Tucker, A. W. (2013). Nonlinear programming. In *Traces and emergence of nonlinear programming.* (pp. 247-258).

[124] Chaudhari, H. A., Lin, S., & Linda, O. (2020). A general framework for fairness in multistakeholder recommendations. *arXiv preprint arXiv:2009.02423*.

[125] Liu, H., & Grigas, P. (2021). Risk bounds and calibration for a smart predict-then-optimize method. *Advances in Neural Information Processing Systems*, *34*. (pp. 22083-22094).

[126] Li, Y., Chen, H., Xu, S., Ge, Y., Tan, J., Liu, S., & Zhang, Y. (2023). Fairness in Recommendation: Foundations, Methods, and Applications. *ACM Transactions on Intelligent Systems and Technology*, *14*(5). (pp. 1-48).

[127] Patro, G. K., Porcaro, L., Mitchell, L., Zhang, Q., Zehlike, M., & Garg, N. (2022). Fair ranking: a critical review, challenges, and future directions. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency.* (pp. 1929-1942).

[128] Kotary, J., Fioretto, F., Van Hentenryck, P., & Wilder, B. (2021). End-to-end constrained optimization learning: A survey. *arXiv preprint arXiv:2103.16378*.

[129] Yang, K., & Stoyanovich, J. (2017). Measuring fairness in ranked outputs. In *Proceedings of the 29th International Conference on Scientific And Statistical Database Management* (pp. 1-6).

[130] Kamishima, T., Akaho, S., Asoh, H., & Sakuma, J. (2014). Correcting Popularity Bias by Enhancing Recommendation Neutrality. In *Poster Proceedings of the 8th ACM Conference on Recommender Systems, RecSys 2014*.

[131] Singh, A., Kempe, D., & Joachims, T. (2021, August 06). Comment: "Response to important issues raised in your review." *Fairness in Ranking under Uncertainty | OpenReview.* Retrieved from https://openreview.net/forum?id=7wunGXQoC27&noteId=RgJyg6Bc2ia

[132] Wang, Y., Wang, L., Li, Y., He, D., & Liu, T. Y. (2013). A theoretical analysis of NDCG type ranking measures. In *Conference on learning theory* (pp. 25-54).

[133] Harper, F. M., & Konstan, J. A. (2015). The MovieLens datasets: History and context. *ACM Transactions On Interactive Intelligent Systems (TIIS), 5(4)*. (pp. 1-19).

[134] Banik, R. The Movies Dataset. (2017). Retrieved from: www.kaggle.com/datasets/rounakbanik/the-movies-dataset

[135] Song, B., Yang, X., Cao, Y., & Xu, C. (2018, October). Neural collaborative ranking. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (pp. 1353-1362).

[136] Pitoura, E., Stefanidis, K., & Koutrika, G. (2022). Fairness in rankings and recommendations: an overview. *The VLDB Journal*, 1-28.

[137] Donti, P., Amos, B., & Kolter, J. Z. (2017). Task-based end-to-end model learning in stochastic optimization. *Advances in neural information processing systems*, *30*.

[138] Wang, K., Wilder, B., Perrault, A., & Tambe, M. (2020). Automatically learning compact quality-aware surrogates for optimization problems. *Advances in Neural Information Processing Systems*, *33*, 9586-9596.

[139] Mao, W., Liu, C., Huang, Y., Zu, Z., Harshvardhan, M., Wang, L., & Zheng, B. (2023, August). End-to-End Inventory Prediction and Contract Allocation for Guaranteed Delivery Advertising. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (pp. 1677-1686).

[140] Perron, L., & Furnon, V. (2019). Or-tools. *Google.[Online]. Available: https://developers. google. com/optimization*.

Appendix A - Fairness Literature Summary

Title	Date	Area	Target	Subj.	Gran.	Opt. Obj.	Method	Approach	Type	Datasets	Brief Description
Ranking with fairness constraints.	April 22, 2017	Fair Ranking	Group	Item	Single	Treatment	Post Processing	COpt; IP	Slot-wise re-ranking	N/A	Proposes constrained ranking maximisation with multiple sensitive attributes.
Fa* ir: A fair top-k ranking algorithm	November 6, 2017	Fair Ranking	Individual	Item	Amortized	Treatment	Post Processing	Greedy Algorithm	Slot-wise re-ranking	German Credit	Maximises ranking utility with group fairness constraint by algorithmic re-ranking with two priority queues for one multinary attribute.
Beyond parity: Fairness objectives for collaborative filtering.	November 30, 2017	Fair RecSys	Group	User	Amortized	Impact	In Processing	Pointwise; Direct Regularization; NCF	Regularisation	MovieLens 1M	Uses fairness metrics (e.g., value fairness) as fair regularisation.
Balanced Neighborhoods for Multi-sided Fairness in Recommendation	February 24, 2018	Fair RecSys	Group	User & Item	Amortized	Treatment	In Processing	Pointwise; Indirect Regularization; SLIM	Regularisation	MovieLens 1M	Adds fairness regularisation to SLIM
Equity of attention: Amortizing individual fairness in rankings.	May 4, 2018	Fair Ranking	Individual	Item	Amortized	Treatment	Post Processing	COpt; IP	User-wise re-ranking	AirBnB, Stack Exchange	Ensures amortised fairness through integer linear programming.
Fairness of exposure in rankings.	August 19, 2018	Fair Ranking	Group	Item	Single	Impact	Post Processing	COpt; LP; Doubly Stochastic Matrix	User-wise re-ranking	Job Seeker, News Recommender	Solves linear programming from the perspective of stochastic rankings for one binary attribute.
Multistakeholder Recommendation with Provider Constraints	September 27, 2018	Multi Stakeholder RecSys	Group	Item	Amortized	Treatment	Post Processing	COpt; IP	Global-wise re-ranking	MovieLens 100K	0-1 integer programming with global provider constraints.
Reducing disparate exposure in ranking: A learning to rank approach.	May 22, 2018	Fair Learning to Rank	Group	Item	Single	Treatment	In Processing	Listwise; Direct Regularisation	Regularisation	W3C Experts, Engineering Students, Law Students	Extends ListNet with a listwise fairness objective that reduces the extent to which protected elements receive less exposure; one binary attribute.
Policy Learning for Fairness in Ranking	February 26, 2019	Fair Learning to Rank	Individual	Item	Single	Treatment	In Processing	Fair-PG-Rank; Policy gradient approach	Reinforcement Learning	Yahoo! LTR, German Credit	Implements policy-gradient for ERM procedure to directly optimise any IR utility metric and a wide range of fairness criteria; one binary attribute.
Fairness in Recommendation Ranking through Pairwise Comparisons	March 2, 2019	Fair Recommender Systems	Group	Item	Single	Impact	In Processing	Pairwise; Indirect Regularisation; MLP	Regularisation	Online	Adds pairwise fairness regularisation based on randomised experiments.
Fairness-Aware Ranking in Search & Recommendation Systems with Application to LinkedIn Talent Search	July 24, 2019	Fair Ranking	Group	Item	Single	Treatment	Post Processing	Greedy Algorithm	Slot-wise re-ranking	Synthetic	Improves multiple group fairness by interval constrained sorting.
FairRec: Two-Sided Fairness for Personalized Recommendations in Two-Sided Platforms	April 20, 2020	Multi Stakeholder RecSys	Individual	Joint	Amortized	Treatment	Post Processing	Greedy Algorithm	Global-wise re-ranking	Google Local; Last FM	Proposes a re-ranking method for both user fairness and item fairness.

Figure 14: Part 1, fairness literature summary table, expands on the work from [81].

Title	Date	Area	Target	Subj.	Gran.	Opt. Obj.	Method	Approach	Type	Datasets	Brief Description
FairRec: Two-Sided Fairness for Personalized Recommendations in Two-Sided Platforms	April 20, 2020	Multi Stakeholder RecSys	Individual	Joint	Amortized	Treatment	Post Processing	Greedy Algorithm	Global-wise re-ranking	Google Local; Last FM	Proposes a re-ranking method for both user fairness and item fairness.
Controlling Fairness and Bias in Dynamic Learning-to-Rank	May 29, 2020	Fair RecSys	Group	Item	Amortized	Treatment & Impact	Post Processing	Unbiased click relevance IPS	Slot-wise re-ranking	MovieLens 20M	Ensures fairness in dynamic learning to rank through p-controller.
TFROM: A two-sided fairness-aware recommendation model for both customers and providers	April 19, 2021	Multi Stakeholder RecSys	Group & Individual	Joint	Amortized	Treatment	Post Processing	Greedy Algorithm	Global-wise re-ranking	Google Local	Proposes a two-sided fairness-aware re-ranking algorithm.
Policy-Gradient Training of Fair and Unbiased Ranking Functions	July 11, 2021	Fair Learning to Rank	Group	Item	Amortized	Treatment & Impact	In Processing	FULTR; Policy gradient approach with unbiased data	Policy Gradient	MSLR, German Credit	Applies policy-gradient for ERM to unbiased data.
End-to-end Learning for Fair Ranking Systems	November 21, 2021	Fair Learning to Rank	Group	Item	Single	Treatment	In Processing	SPOFR; decision focused learning	Decision-Focused Learning	MSLR, German Credit	Incorporates constrained optimization programs using SPO into the training loop.

Figure 15: Part 2, fairness literature summary table, expands on the work from [81].

Appendix B - FairRec Algorithm

Algorithm 1 *FairRec* (U, P, k, V)

Input: Set of customers $U = [m]$, set of distinct products $P = [n]$, recommendation set size k (such that $k < n$ and $n \leq k \cdot m$), and the relevance scores $V_u(p)$.
Output: A two-sided fair recommendation.

Initialize allocation $\mathcal{A}^0 = (A_1^0, \ldots, A_m^0)$ with $A_i^0 \leftarrow \emptyset$ for each customer $i \in [m]$.

First Phase:
Fix an (arbitrary) ordering of the customers $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(m))$.
Initialize set of feasible products $F_u \leftarrow P$ for each $u \in [m]$.
Set $\ell \leftarrow \left\lfloor \frac{m \times k}{n} \right\rfloor$ denoting number of copies of each product.
Initialize each component of the vector $S = (S_1, \ldots, S_n)$ with $S_j \leftarrow \ell$, $\forall j \in [n]$, this stores the number of available copies of each product.
Set $T \leftarrow \ell \times n$, total number of items to be allocated.
$[\mathcal{B}, F, x] \leftarrow$Greedy-Round-Robin$(m, n, S, T, V, \sigma, F)$.
Assign $\mathcal{A} \leftarrow \mathcal{A} \cup \mathcal{B}$.

Second Phase:
Set $\Lambda = |A_{\sigma((x) \pmod m)+1)}|$ denoting the number of items allocated to the customer subsequent to x, according to the ordering σ.
if $\Lambda < k$ **then**
    Update each component of the vector $S = (S_1, \ldots, S_n)$ with the value m in order to allow allocating any product to any customer.
    Set $T \leftarrow 0$.
    if $x < m$ **then**
        Set $\sigma'(i) \leftarrow \sigma((i + x - 1) \pmod m + 1)$ for all $i \in [m]$.
        $\sigma \leftarrow \sigma'$.
        $T \leftarrow (m - x)$.
        Update $\Lambda \leftarrow \Lambda + 1$.
    end if
    $T \leftarrow T + m(k - \Lambda)$ total number of items to be allocated.
    $[\mathcal{C}, F, x] \leftarrow$Greedy-Round-Robin$(m, n, S, T, V, \sigma, F)$.
    Assign $\mathcal{A} \leftarrow \mathcal{A} \cup \mathcal{C}$.
end if
Return $\mathcal{A}$.

Figure 16: FairRec algorithm 1, taken from [63].

Algorithm 2 Greedy-Round-Robin $(m, n, S, T, V, \sigma, F)$

Input : Number of customers m, number of producers n, an array with number of available copies of each product S, total number of available products $T > 0$, relevance scores $V_u(p)$ and feasible product set F_u for each customer, and an ordering σ of $[m]$.
Output: An allocation of T products among m customers, the residual feasible set F_u and the last allocated index x.

1: Initialize allocation $\mathcal{B} = (B_1, \ldots, B_m)$ with $B_i \leftarrow \emptyset$ for each customer $i \in [m]$.
2: Initiate $x \leftarrow m$.
3: Initiate round $r \leftarrow 0$.
4: **while** true **do**
5: Set $r \leftarrow r + 1$.
6: **for** $i = 1$ to m **do**
7: Set $p \in \underset{p' \in F_{\sigma(i)}:(S_p \neq 0)}{\arg\max} V_{\sigma(i)}(p')$
8: **if** $p == \emptyset$ **then**
9: Set $x = i - 1$ only if $i \neq 1$.
10: **go to** Step 22.
11: **end if**
12: Update $B_{\sigma(i)} \leftarrow B_{\sigma(i)} \cup p$.
13: Update $F_{\sigma(i)} \leftarrow F_{\sigma(i)} \setminus p$.
14: Update $S_p \leftarrow S_p - 1$.
15: Update $T \leftarrow T - 1$.
16: **if** $T == 0$ **then**
17: $x = i$.
18: **go to** Step 22.
19: **end if**
20: **end for**
21: **end while**
22: Return $\mathcal{B} = (B_1, \ldots, B_m)$, $F = (F_1, \ldots, F_m)$ and index x.

Figure 17: FairRec algorithm 2, taken from [63].