Counterfactual Explanations for Recommendation Bias

A Thesis

submitted to the designated
by the Assembly
of the Department of Computer Science and Engineering
Examination Committee

by

Leonidas Zafeiriou

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN DATA AND COMPUTER
SYSTEMS ENGINEERING

WITH SPECIALIZATION
IN DATA SCIENCE AND ENGINEERING

University of Ioannina
School of Engineering
Ioannina 2023

Acknowledgements

I would like to express my gratitude to my supervisor Assoc. Professor Panayiotis Tsaparas and Professor Evaggelia Pitoura for their continuous support and guidance. Working with them was a valuable learning experience. I appreciate our collaborative relationship in the process of completing this thesis.

Table of Contents

List of Figures

Abstract

Leonidas Zafeiriou, M.Sc. in Data and Computer Systems Engineering, Department of Computer Science and Engineering, School of Engineering, University of Ioannina, Greece, 2023.
Counterfactual Explanations for Recommendation Bias.
Advisor: Panayiotis Tsaparas, Assoc. Professor.

Today, we rely heavily on automated recommendation algorithms for assisting us in making several decisions, such as the content we consume, the items we buy, or the careers we pursue. These algorithms use sophisticated machine learning techniques that are trained on large quantities of user interaction data. As a result they incorporate various biases in their recommendations, where certain groups of users or items are treated differently. Understanding the recommender biases is important in monitoring the health of the recommendation system and achieving fairness. However, given the complexity of recommender algorithms, this is becoming increasingly difficult. To address this issue there is a strong research movement towards producing different types of explanations for the behavior of the algorithms. One type of explanations is counterfactual explanations where we look for a small number of changes in the input data that will achieve a desired change in the output of the algorithm on a specific data instance, e.g., increase the score of the recommender for a specific user-item pair. In this work, we consider counterfactual explanations for recommendation bias. Given that bias is defined with respect to groups of users and items instead of specific user-item pairs, we generalize the definition of counterfactual explanations to handle this case. We then consider a random-walk based recommender, and we propose algorithms for computing the counterfactual explanations. Our algorithms are efficient and they can be applied to large datasets. We perform an experimental evaluation of our algorithms using both real and synthetic data.

Εκτεταμενη Περιληψη

Λεωνίδας Ζαφειρίου, Δ.Μ.Σ. στη Μηχανική Δεδομένων και Υπολογιστικών Συστημάτων, Τμήμα Μηχανικών Η/Υ και Πληροφορικής, Πολυτεχνική Σχολή, Πανεπιστήμιο Ιωαννίνων, 2023.
Εξηγήσεις με Αντιπαράδειγμα για Συστήματα Συστάσεων με Προκαταλήψεις.
Επιβλέπων: Παναγιώτης Τσαπάρας, Αναπλ. Καθηγητής.

Σήμερα, βασιζόμαστε σε μεγάλο βαθμό σε αυτόματους αλγόριθμους συστάσεων για να μας βοηθήσουν στη λήψη αποφάσεων σε πολλά θέματα, όπως τα περιεχόμενα που καταναλώνουμε, τα αντικείμενα που αγοράζουμε ή τις καριέρες που ακολουθούμε. Αυτοί οι αλγόριθμοι χρησιμοποιούν προηγμένες τεχνικές μηχανικής μάθησης που εκπαιδεύονται σε μεγάλους όγκους δεδομένων αλληλεπίδρασης των χρηστών. Ως αποτέλεσμα, ενσωματώνουν διάφορες προκαταλήψεις στις συστάσεις τους, όπου ορισμένες ομάδες χρηστών ή αντικειμένων αντιμετωπίζονται διαφορετικά. Παρόλο που αυτές οι προκαταλήψεις είναι ως ένα σημείο απαραίτητες ώστε να δοθούν εξατομικευμένες συστάσεις, μπορούν να οδηγήσουν σε άνιση αντιμετώπιση ορισμένων ομάδων.

Η κατανόηση των προκαταλήψεων των συστημάτων συστάσεων είναι σημαντική για την παρακολούθηση της ορθής λειτουργίας του συστήματος συστάσεων και την επίτευξη της δικαιοσύνης. Ωστόσο, λόγω της πολυπλοκότητας των αλγορίθμων συστάσεων, αυτό γίνεται όλο και πιο δύσκολο. Για να αντιμετωπιστεί αυτό το ζήτημα, υπάρχει ένα ισχυρό κίνημα έρευνας προς την παραγωγή διαφορετικών τύπων εξηγήσεων για τη συμπεριφορά των αλγορίθμων. Ένας τύπος εξηγήσεων είναι οι εξηγήσεις με αντιπαράδειγμα, όπου αναζητούμε μια μικρή αλλαγή στα δεδομένα εισόδου που θα επιτύχει μια επιθυμητή αλλαγή στην έξοδο του αλγορίθμου σε μια συγκεκριμένη περίπτωση δεδομένων, για παράδειγμα, αύξηση του σκορ του συστηματος συστάσεων για ένα συγκεκριμένο ζεύγος χρήστη-αντικειμένου.

Στην παρούσα εργασία, εξετάζουμε εξηγήσεις με αντιπαράδειγμα για τις προκαταλήψεις σε συστήματα συστάσεων. Δεδομένου ότι οι προκαταλήψεις καθορίζονται σε σχέση με ομάδες χρηστών και αντικειμένων αντί για συγκεκριμένα ζεύγη χρηστή-αντικειμένου, γενικεύουμε τον ορισμό των εξηγήσεων με αντιπαράδειγμα για να αντιμετωπίσουμε αυτήν την περίπτωση. Εξετάζουμε διαφορετικούς τύπους εξηγήσεων. Αρχικά, εξετάζουμε τους μεμονωμένους χρήστες και αναζητούμε εξηγήσεις γιατί ένας χρήστης δεν λαμβάνει αρκετές προτάσεις για μια συγκεκριμένη κατηγορία αντικειμένων. Επεκτείνουμε αυτές τις εξηγήσεις στην περίπτωση που έχουμε μια ομάδα χρηστών αντί για έναν μεμονωμένο χρήστη. Στη συνέχεια, εξετάζουμε τα μεμονωμένα αντικείμενα και αναζητούμε εξηγήσεις γιατί δεν προτείνονται σε μια συγκεκριμένη ομάδα χρηστών. Επεκτείνουμε και πάλι τις εξηγήσεις αυτές για την περίπτωση όπου έχουμε μία ομάδα αντικειμένων αντί για ένα μεμονωμένο αντικείμενο.

Θεωρούμε ένα σύστημα συστάσεων τυχαίου περίπατου βασισμένου σε γραφήματα και προτείνουμε αλγορίθμους για τον υπολογισμό εξηγήσεων με αντιπάραδειγμα. Οι αλγόριθμοι μας χρησιμοποιούν εργαλεία Γραμμικής Άλγεβρας για τον αποδοτικό υπολογισμό της αλλαγής της πόλωσης του συστήματος συστάσεων και μπορούν να εφαρμοστούν και σε μεγάλα σύνολα δεδομένων. Κάνουμε πειραματική αξιολόγηση των αλγορίθμων μας χρησιμοποιώντας ένα σύνολο ταινιών, καθώς και συνθετικά δεδομένα. Τα πειράματά μας μελετάνε την δυσκολία εύρεσης εξηγήσεων για διάφορες περιπτώσεις και παρέχουν κατανόηση των χαρακτηριστικών του συνόλου δεδομένων τα οποία επηρεάζουν τις εξηγήσεις.

Chapter 1

Introduction

Today, we rely heavily on automated recommendation algorithms for assisting us in making several decisions, such as the content we consume, the items we buy, or the careers we pursue. These algorithms use sophisticated machine learning techniques that are trained on large quantities of user interaction data. As a result, they incorporate various *biases* in their recommendations, where certain groups of users or items are treated differently. Although these biases are to some extent integral to the algorithms in order to make personalized recommendations, they can also lead to unfair treatment of sensitive groups.

Fairness and bias in recommendations is a problem that has received significant attention in the past years [1, 2]. Different definitions of fairness have been adopted, depending on whether fairness is defined with respect to consumers or producers [3]. In most definitions, we assume that there are groups of users and/or items, defined based on sensitive attributes such as gender, religion or age for users, and type of content for items. The recommender should treat the groups in a fair manner, e.g., producing recommendations of equal quality for two user groups, or representing proportionally the items in different item groups. When this is not achieved, we consider the recommender to be biased.

Understanding recommendation biases is important in monitoring the health of the recommendation system and ensuring fairness. However, given the complexity of recommendation algorithms, this is becoming increasingly difficult. To address the algorithmic complexity of the "black box", there is a strong research movement towards producing different types of explanations for the behavior of algorithms, including recommendation algorithms [4]. One type of explanations is counterfactual explanations [5], where we look for a small number of changes in the input data that will achieve a desired change in the output of the algorithm on a specific data instance, e.g., change the classification of a data point.

In this work, we consider counterfactual explanations for recommendation bias. Previous work on counterfactual explanations for fairness focused on explaining the decisions of the recommender for specific user-item pairs [6, 7, 8]. Given that bias is defined with respect to groups of users and items instead of specific user-item pairs, we need to generalize the definition of counterfactual explanations to handle this case. We consider different types of explanations. First, we consider individual users, and we seek explanations as to why a user does not get enough recommendations from a specific item category. We extend these explanations to the case where we have a group of users instead of an individual user. We then consider individual items and look for explanations as to why they do not get recommended to a specific group of users. Again, we extend these explanations to the case where we have an item category rather than a single item.

We consider a graph-based random walk recommender, and we propose algorithms for computing the counterfactual explanations. Our algorithms exploit Linear Algebra tools to efficiently estimate the effect of a change to the bias of the recommender, and it can be applied to large datasets. We perform an experimental evaluation of our algorithms using a real movies dataset, as well as synthetic data. Our experiments study the hardness of producing explanations for different cases, and provide understanding of the dataset characteristics that affect the explanations.

In summary, in this work we make the following contributions:

- We consider and formalize the novel problem of counterfactual explanations for different types of recommendation bias.
- We propose efficient algorithms for computing counterfactual explanations for a random walk recommender that can scale on large datasets.
- We evaluate quantitatively and qualitatively our algorithms on real and synthetic datasets.

The rest of the work is structured as follows. In Chapter 2 we provide the definitions for our problems. In Chapter 3 we present methods for computing the effect of a counterfactual, and algorithms for producing counterfactuals. In Chapter 4 we present our experimental evaluation. Chapter 5 presents the related work, and Chapter 6 concludes present work.

Chapter 2

Definitions

We are given as input a set of users $\mathcal{U}$ and a set of items $\mathcal{I}$ and a user-item matrix D with the preferences of the users over the items. We also have a recommender R_D that is trained on the matrix D, which given a pair $(u, i) \in \mathcal{U} \times \mathcal{I}$, it outputs a score $R_D(u, i)$ that is the estimation of the preference of user u for the item i.

We assume that we can define groups of users and items based on the properties of the users and items respectively. For example, we may partition users into male and female based on gender, or we may define groups of users based on age, or on residence. Similarly, if the items are movies, we may define subsets of movies based on genre, or on release date. We are interested in defining biases that the recommender may have towards specific groups of items or users, and provide explanations for them.

We first consider the bias of the recommender in the estimated ratings for an individual user towards a specific group of items. For example in the user-movie scenario, we want to explain why the estimated ratings for a specific user are on average higher for the Action movies, compared to those for Romance movies. Formally, let u be a specific user, let $I \subset \mathcal{I}$ denote a subset of the items that belong to a specific group, and let $\overline{I} = \mathcal{I} \setminus I$ denote the items not in the group. We define $R_D(u, I) = \frac{1}{|I|} \sum_{i \in I} R_D(u, i)$ to be the average estimated score of the recommender for user u for the items in I. $R_D(u, \overline{I})$ is defined similarly. We define the bias of recommender R_D in the recommendations for user u towards group I as:

$$B_{R_D}(I|u) = \frac{R_D(u, I)}{R_D(u, \overline{I})}$$

When $B_{R_D}(I|u) > 1$ the recommender is biased in favor of I in the scores it produces for u, meaning that it estimates higher average score for the items in group I than the rest. When $B_{R_D}(I|u) < 1$ the recommender is biased against the group I.

We seek explanations for the bias of the recommender in the recommendations to an individual user u towards group I. We assume that I is being treated unfairly ($B_{R_D}(I|u) < 1$), and thus we want to increase $B_{R_D}(I|u)$. The explanations we will produce will be in the form of *counterfactuals*: changes in the ratings D_u of user u that will result in an increase $B_{R_D}(I|u)$. We refer to these explanations as *individual user bias explanations*. We thus have the following definition:

Definition 2.1 (Individual User Bias Explanation). Given user-item preference matrix D, a recommender R_D, a user $u \subset \mathcal{U}$, a group of items I, and a target value θ, an individual user bias explanation is the minimum subset $E_u \subseteq D_u$ such that if deleted from D, $B_{R_{D|E_u}}(I|u) \geq (1+\theta)\ B_{R_D}(I|u)$.

We can extend the definition of bias to the case that we have a group of users instead of a single user. In the user-movie example, we want to explain why the average ratings of male users are on average lower for Romance movies than for Action movies. Let $U \subset \mathcal{U}$ denote a subset of the users that belong to a specific group. For a group of items I, we define $R_D(U, I) = \frac{1}{|U|}\sum_{u \in U} R_D(u, I)$ to be the average over the users in U of the average estimated score of the recommender for the items in I. $R_D(U, \overline{I})$ is defined similarly. We define the bias of recommender R_D in the recommendations for group U towards group I as:

$$B_{R_D}(I|U) = \frac{R_D(U, I)}{R_D(U, \overline{I})}$$

Again, $B_{R_D}(I|u) > 1$ means that the recommender is biased in favor of I in the scores it produces for the user group U, while $B_{R_D}(I|u) < 1$ means that the recommender is biased against the item group I.

We seek again counterfactual explanations for the bias of the recommender in the recommendations to an user group U against item group I. Let D_U denote the set of ratings from users in U. We look for changes in the ratings D_u of user u that will result in an increase $B_{R_D}(I|U)$. We refer to these explanations as *individual user bias explanations*. We thus have the following definition:

Definition 2.2 (User Group Bias Explanation). Given user-item preference matrix D, a recommender R_D, a group of users $U \subset \mathcal{U}$, a group of items I, and a target value θ, a user group bias explanation is the minimum subset $E_U \subseteq D_U$ such that if deleted from D, $B_{R_{D|E_U}}(I|U) \geq (1+\theta)\ B_{R_D}(I|U)$.

We then turn our attention to the item side, and we consider the bias of the recommender in the estimated ratings that an individual item i receives from a group of users U. In our user-movie example, we look at a specific movie, and we want to explain, why this movie receives on average lower ratings from male users than from female users. For example, why is it the case that the movie ”The English Patient” appeals less to male than to female users?

Formally, let i be a specific item, let $U \subset \mathcal{I}$ denote a subset of the users that belong to a specific group, and let $\overline{U} = \mathcal{I} \setminus U$ denote the users not in the group. We define $R_D(U,i) = \frac{1}{|U|}\sum_{u \in U} R_D(u,i)$ to be the average estimated score of the recommender for group U for the item i. $R_D(\overline{U},i)$ is defined similarly. We define the bias of recommender R_D in the recommendations for item i from group U as:

$$B_{R_D}(U|i) = \frac{R_D(U,i)}{R_D(\overline{U},i)}$$

When $B_{R_D}(U|i) > 1$ the recommender is biased in favor of U in the recommendations for item i, meaning that it estimates higher average score for the users in group U than the rest for i. When $B_{R_D}(I|u) < 1$ the recommender is biased against the group U.

We seek again counterfactual explanations for the bias of the recommender in the recommendations for an individual i u from a group I. We assume that U is being treated unfairly ($B_{R_D}(U|i) < 1$), and thus we want to increase $B_{R_D}(U|i)$. We look for changes in the ratings D_U of the group U that will result in an increase $B_{R_D}(U|i)$. We refer to these explanations as *individual item bias explanations*. We thus have the following definition:

Definition 2.3 (Individual Item Bias Explanation). Given user-item preference matrix D, a recommender R_D, an item $i \subset \mathcal{I}$, a group of users U, and a target value θ, an individual item bias explanation is the minimum subset $E_U \subseteq D_U$ such that if deleted from D, $B_{R_{D|E_U}}(U|i) \geq (1+\theta)\ B_{R_D}(U|i)$.

We will also consider the the bias of the recommender in the estimated ratings that a group of items I receives from a group of users U. In our example, we want to explain why the romance category receives lower scores for males than it receives for females. We define the bias of recommender R_D in the recommendations for item i from group U as:

$$B_{R_D}(U|I) = \frac{R_D(U, I)}{R_D(\overline{U}, I)}$$

As in the case of the item bias explanations we seek changes in D_U that will result in an increase in $B_{R_D}(U|I)$.

Definition 2.4 (Item Group Bias Explanation). Given user-item preference matrix D, a recommender R_D, an item group $I \subset \mathcal{I}$, a group of users U, and a target value θ, an item group bias explanation is the minimum subset $E_U \subseteq D_U$ such that if deleted from D, $B_{R_{D|E_U}}(U|I) \geq (1 + \theta)\ B_{R_D}(U|I)$.

Our definitions of bias and explanations for bias are general, but clearly, the exact explanations depend on the recommendation algorithm R_D. In the next section, we present the recommendation algorithm R_D that we will consider in this work, and an efficient method for computing individual and group explanations for random-walk recommenders.

Chapter 3

Explanations in Graph Recommenders

3.1 The recommendation algorithm

As our recommendation algorithm, we will use the RecWalk algorithm [9]. We view the user-item matrix D as a bipartite graph G with adjacency matrix:

$$A_G = \begin{pmatrix} 0 & D \\ D^T & 0 \end{pmatrix}$$

The RecWalk algorithm estimates the scores for user-item pairs by performing random walks on the graph G. Let $H = Diag(A_G \mathbb{1})^{-1} A_G$ be the transition probability of a simple random walk on the user-item bipartite graph. This transition matrix can also exploit an item mode. Let M_I be an inter-item transition probability matrix that captures relations between items, and define matrix M as:

$$M = \begin{pmatrix} I & 0 \\ 0 & M_I \end{pmatrix}.$$

The overall transition probability matrix of RecWalk is defined as $P = \alpha\, H + (1-\alpha)\, M$ where α captures the relative contribution of each of the two components in the random walk.

To compute recommendations for a user u, we perform a *personalized random walk* rooted on u. At each step the random walk with probability $(1-\gamma)$ transitions according to matrix P, while with probability γ it restarts from node u. For the stationary probability $\mathbf{p}_u$ of the random walk we have:

$$\mathbf{p}_u = \gamma\, \mathbf{e}_u + (1-\gamma)\mathbf{p}_u\, P \tag{3.1}$$

where γ is the jump probability. The estimated rating of item i for user u is given by: $R_D(u,i) = \mathbf{p}_u(i)$.

Note that we can compute the $\mathbf{p}_x$ vectors for all nodes $x \in G$, both users and items, by computing the matrix

$$Q = \gamma\, (I - (1-\gamma)P)^{-1}\,. \tag{3.2}$$

It is easy to see that $\mathbf{p}_x = \mathbf{e}_x Q$, and thus the x-th row of matrix Q holds the $\mathbf{p}_x$ vector.

Given this graph view of the data and the algorithm, the counterfactual explanations that we consider consist of (directed) edges, which if deleted from the bipartite graph would result in the increase of the estimated ratings of a user or a set of users for an item or a set of items, depending on the problem we consider. We view the edges of the bipartite graph as pairs of directed edges, and our explanations will remove edges from the user side towards the item side.

3.2 Estimate the effect of edge removals

We will begin with the estimation of the change of the estimated rating $R_D(u,i)$ for a user-item pair (u,i), when deleting an edge (x,y) from the graph. Note that x may be different from u, and y may be different form i. We want to estimate

$$\Delta\,(u,i,(x,y)) = R_{D|(x,y)}(u,i) - R_D(u,i)$$

We will provide analytical formulas for this computation which we will then use for the different problems we consider in this paper. For simplicity we assume that $\alpha = 0$, but our formulas below can easily be extended to the case that $\alpha \neq 0$.

Recall that $R_D(u,i) = \mathbf{p}_u(i)$ denote the personalized random walk probability of user u for item i. We use $R_{D|(x,y)}(u,i) = \mathbf{p}_u(i|(x,y))$ to denote the probability of user

u for item i after the removal of edge (x, y). We want to estimate $\Delta(u, i, (x, y)) = \mathbf{p}_u(i|(x, y)) - \mathbf{p}_u(i)$.

For the following, we use D_x to denote the outgoing edges from node x in the bipartite graph. This node may be either a user or an item. The vector $\mathbf{p}_u$ is the output of a personalized random walk using the matrix P and it assigns probabilities to both the items and the users in the bipartite graph. Furthermore, note that using the matrix P we can also run personalized random walks for items as well, and estimate a vector $\mathbf{p}_i$ for any item i.

We can prove the following (the proof appears in the appendix):

$$\Delta\left(u, i, (x, y)\right) = \mathbf{p}_u(x)\Lambda(x, i, (x, y)) \tag{3.3}$$

where

$$\Lambda\left(x, i, (x, y)\right) = \frac{\frac{1-\gamma}{\gamma}\left(\frac{1}{|D_x|}\sum_{j \in D_x} \mathbf{p}_j(i) - \mathbf{p}_y(i)\right)}{|D_x| - 1 - \frac{1-\gamma}{\gamma}\left(\frac{1}{|D_x|}\sum_{j \in D_x} \mathbf{p}_j(x) - \mathbf{p}_y(x)\right)} \tag{3.4}$$

This formula can be extended to the case where we remove multiple edges from the node x. We will use $E_x \subset \{(x, y) : y \in D_x\}$ to denote the set of edges removed from x (note that we assume that at least one edge from x remains in the graph). Given that the left endpoint of the edges is fixed to x, abusing the notation, we will also use E_x to denote the set of neighbors of x from which we remove the edges, that is, the set of right endpoints of the removed edges. We can estimate $\Delta(u, i, E_x) = \mathbf{p}_u(i|E_x) - \mathbf{p}_u(i)$ as follows:

$$\Delta(u, i, E_x) = \mathbf{p}_u(x)\Lambda(x, i, E_x) \tag{3.5}$$

where

$$\Lambda(x, i, E_x) = \frac{\frac{1-\gamma}{\gamma}\left(\frac{1}{|D_x|}\sum_{j \in D_x} \mathbf{p}_j(i) - \frac{1}{|E_x|}\sum_{j \in E_x} \mathbf{p}_j(i)\right)}{\frac{|D_x| - |S_x|}{|S_x|} - \frac{1-\gamma}{\gamma}\left(\frac{1}{|D_x|}\sum_{j \in D_x} \mathbf{p}_j(x) - \frac{1}{|E_x|}\sum_{j \in E_x} \mathbf{p}_j(x)\right)} \tag{3.6}$$

Note that Equations 3.3 and 3.4 are special cases of Equations 3.5 and 3.6 when E_x contains a single edge.

Consider now the case where we want to compute explanations for an individual user u, by removing a set of edges E_u from user u to increase the score that the recomender gives to the group of items I. Abusing the notation, we will use $\mathbf{p}_u(I) = \sum_{i \in I} \mathbf{p}_u(i)$ to denote the total probability that the personalized random walk gives to the group I. Therefore, $R(u,i) = \frac{1}{|I|}\mathbf{p}_u(I)$. Let $\Delta(u,I,E_u) = R_{D|E_u}(I|u) - R_D(I|u)$. Using Equations 3.5 and 3.6, we have:

$$\Delta(u,I,E_u) = \mathbf{p}_u(u)\Lambda(u,I,E_u) \tag{3.7}$$

where

$$\Lambda(u,I,E_u) = \frac{\frac{1-\gamma}{\gamma}\left(\frac{1}{|D_u|}\sum_{j\in D_u}\frac{1}{|I|}\mathbf{p}_j(I) - \frac{1}{|E_u|}\sum_{j\in E_u}\frac{1}{|I|}\mathbf{p}_j(I)\right)}{\frac{|D_u|-|E_u|}{|E_u|} - \frac{1-\gamma}{\gamma}\left(\frac{1}{|D_u|}\sum_{j\in D_u}\mathbf{p}_j(u) - \frac{1}{|E_u|}\sum_{j\in E_u}\mathbf{p}_j(u)\right)} \tag{3.8}$$

Consider now a group of users U and an item i. The effect of removing a set of edges E_x from a node x can be estimated as

$$\Delta(U,i,E_x) = \mathbf{p}_U(x)\Lambda(x,i,E_x) \tag{3.9}$$

where $\mathbf{p}_U(x) = \frac{1}{U}\sum_{u\in U}\mathbf{p}_u(x)$ and $\Lambda(x,i,E_x)$ is computed as in Equation 3.4.

When considering a group of users U and a group of items I, the effect of removing a set of edges E_x from a node x can be estimated as

$$\Delta(U,I,E_x) = \mathbf{p}_U(x)\Lambda(x,I,E_x) \tag{3.10}$$

where $\Lambda(x,I,E_x)$ is computed as in Equation 3.8.

3.3 Computing the explanations

We will now describe how we can efficiently compute the different terms that appear in the Equations we presented above (and specifically, Equations 3.5 and 3.6, that are the basis for the rest).

We consider the case where we remove a set of edges E_x from a node x and we want to estimate $\Delta(u,I,E_x) = \mathbf{p}_u(x)\Lambda(x,I,E_x)$ for some node u. The formulas for this computation require the computation of the quantities $\mathbf{p}_i(x)$ for every node $i \in G$, and $\mathbf{p}_i(I)$ for every node $i \in G$.

The naive implementation would be to estimate all personalized vectors that we need. We can estimate the quantities efficiently using a single *absorbing random walk*. The cost of the absorbing random walk is essentially the same as that of executing a Pagerank algorithm, thus making the estimation very efficient.

An absorbing random walk is walk where some nodes, or states, are *absorbing*. A node, or state, is absorbing when the random walk can only enter the node and there is no transition out of the node. We create our absorbing random walk by augmenting the graph G with the addition of four absorbing nodes:

- Absorbing node A_x: Only node x links to this absorbing node.
- Absorbing node $A_{\mathcal{U}}$: All the user nodes in $\mathcal{U}$, except for x, link to this absorbing node.
- Absorbing node A_I: All the items in I link to this absorbing node.
- Absorbing node $A_{\overline{I}}$: All the items in $\overline{I}$ link to this absorbing node.

Note that each transient node (non-absorbing) is connected to a single absorbing state. Each absorbing node A_Z, corresponds to a distinct subset $Z \subset V$ of the nodes of the graph G, as defined above. We set the transition probability from any transient node to the corresponding absorbing node to be γ, the restart probability of the personalized random walk. With probability $1-\gamma$ the random walk transitions to another transient state using the transition matrix P.

In this absorbing random walk we are interested in estimating the *absorption probability* $\alpha_i(A_Z)$ for any node i in the graph, which is the probability that a random walk that starts form node i will be absorbed at absorbing node A_Z. This can be estimated efficiently for all nodes $i \in G$ and all absorbing states A_Z with a simple iterative procedure. The key observation is that $\alpha_x(A_Z) = p_x(Z)$, that is, the absorption probability from node x to an absorbing state A_Z is equal to the probability allocated to the set Z by the personalized random walk rooted at x.

The iterative procedure is as efficient as running a Pagerank algorithm. For every node i we maintain a vector with the four absorbing probabilities we want to compute, one for each absorbing node. These are initialized to zero. For a node i, and an absorbing node A_Z, if $i \notin Z$, then $\alpha_i(A_Z) = (1-\gamma)\sum_{j \in D_i} P_{ij}\alpha_i(A_Z)$. If $i \in Z$, $\alpha_i(A_Z) = \gamma + (1-\gamma)\sum_{j \in D_i} P_{ij}\alpha_i(A_Z)$. Repeating until convergence gives the desired probabilities.

3.4 Algorithms for computing bias explanations

3.4.1 Individual user explanations.

In the case of individual user explanations to a user u, we want to explain why for user u the scores of the recommender R for group I are lower than those for group $\overline{I}$. To find these explanations we look for the edges $E_u \subset D_u$ emanating from u whose removal will maximize $\Delta(u, I, E_u)$. We use a *greedy* algorithm for this task. We incrementally build the set E_u, each time adding the edge (u, v) that maximizes the gain $\text{gain}(u, v) = \Delta(u, I, E_u \cup \{(u, v)\}) - \Delta(u, I, E_u \cup \{(u, v)\})$. Note that we can implement the Greedy algorithm very efficiently. We compute for every node i in the graph the quantities $\mathbf{p}_i(u)$ and $\mathbf{p}_i(I)$, using the absorbing random walk. This computation is done once, at the beginning of the algorithm. Then at any iteration of the algorithm we can compute $\text{gain}(u, v)$ with simple mathematical operations using Equation 3.7. We will refer to this algorithm as Greedy.

For comparison, we will also consider a simpler version of the greedy algorithm that computes $\Delta(u, I, (u, v))$ for each edge (u, v), sorts the edges in decreasing order of the Δ-values, and select them in that order. This algorithm is more efficient as it makes only one computation initially.

3.4.2 User group explanations.

In the case of explanations to a user group, we want to explain why for the user group U the scores of the recommender R for the item group I are lower than those for the complement item group $\overline{I}$. The explanation E_U consists of a set of changes in the ratings of the users in U that correct the bias of the recommender. We will consider three algorithms for constructing the explanations, each leading to a qualitatively different type of explanations.

The first algorithm looks for the best set of edges from U to remove. Note that it is relatively trivial to show that we only need to consider edges (u, v) to the complement item group $\overline{I}$, that is, $v \in \overline{I}$; otherwise, we decrease $R(U, I)$. The algorithm computes the value $\Delta(U, I, (u, i))$ for each edge (u, i), where $u \in U$, and $i \in \overline{I}$, it sorts the edges, and selects the top ones that achieve the target bias goal. We will refer to this algorithm as EdgeExplain.

The second algorithm builds the explanation by selecting *users* from U, and removing all their edges to the complement group $\overline{I}$. The explanation in this case is a set of users who, if disconnected from the complement item group $\overline{I}$ it will correct the bias of the recommender. To compute the explanation, for each user $u \in U$, let $E_u(\overline{I})$ denote the set of edges from u to the group $\overline{I}$. The algorithm estimates $\Delta(u, I, E_u(\overline{I}))$ for all users $u \in U$, sorts them according to the Δ values, and returns the top ones that achieve the target bias goal. We will refer to this algorithm as UserExplain.

The third algorithm builds the explanation by selecting *items* from the complement group $\overline{I}$, and removing all edges from the group U. The explanation in this case is a set of items, such that if they disconnected from group U, it will correct the bias of the recommender. For an item $i \in \overline{I}$, we approximate the effect of its removal from group U by computing $\Delta(U, I, i) = \sum_{u \in U} \Delta(U, I, (u, i))$. We sort the items according to the Δ values, and return the top ones that achieve the target bias goal. We will refer to this algorithm as ItemExplain.

3.4.3 Individual Item Explanations.

In the case of individual item explanations, we want to explain why for a specific item, the recommendation algorithm estimates lower scores from group U than $\overline{U}$. The explanations consist of edges from the user group U. For an item i, it is relatively easy to see that the best edges to remove are from the users that have rated the item i, that is, users in $D_i \subseteq U$. For each edge (x, y), $x \in D_i$, $y \neq i$, we compute $\Delta(U, i, (x, y))$. We sort the edges according to these values, and we return the top edges that achieve the target bias value.

3.4.4 Item Group Explanations.

In the case of item group explanations, we want to explain why for a group of items, the recommendation algorithm estimates lower scores from group U than $\overline{U}$. For this case, we adopt the three different types of explanations we described for the user group explanations, and the corresponding algorithms.

Chapter 4

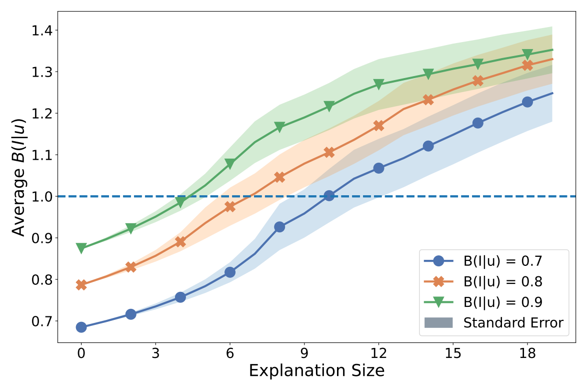
Experiments

We now evaluate our algorithms for producing explanations for the different recommendation bias. Depending on the type of bias we consider we will consider different approaches for selecting explanation edges. The goal of the experiments is to understand quantitatively and qualitatitively the different explanations we produce.

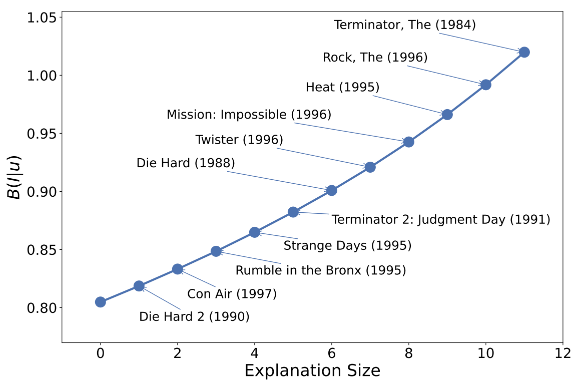
4.1 Datasets

We evaluate our algorithm using both real and synthetic datasets. We will now describe our datasets and their characteristics.

Real Dataset: We use MovieLens 100K Dataset [10] which is a dataset of user ratings on movies. It consists of 100,000 ratings from 943 users on 1682 movies. There is demographic information about the users, and genre information about the movies. For our experiments we used the gender to define groups of users, and the movie



(a) Greedy explanations



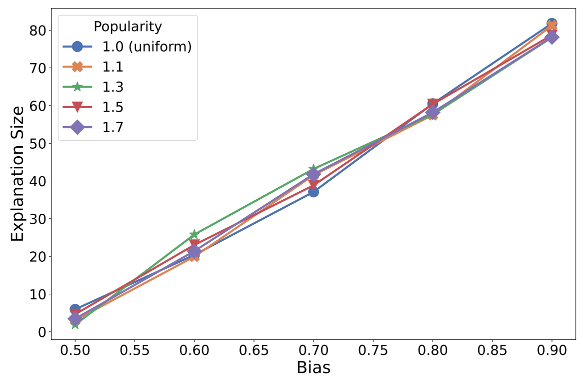
(b) Explanations Example

Figure 4.1: Individual user explanations

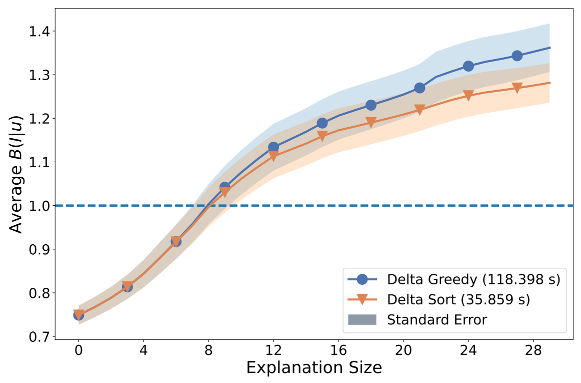
genre for defining groups of movies. We used a subset of the dataset with the movies of Action or Romance genre, exclusively. The resulting dataset consists of all users, 670 males and 273 females and 448 movies, 226 Action and 222 Romance. In our experiments we will denote the group of Males as M and the group of Females as F, and the Romance group as R and the Action group as A.

Synthetic Datasets: We also created synthetic datasets to study our algorithms. All datasets consist of $N_U = 1000$ users and $N_I = 1000$ items. Users are partitioned into two groups, U_0 and U_1, of equal size, and items into two categories, I_0 and I_1, also of equal size. For each user we allocated a constant number of ratings equal to 10% of total number of items, i.e., 100 ratings per user in our case. We introduce bias in the data, where users in U_0 favor items in I_0, and users in U_1 favor items in I_1. We control the bias with a parameter β: For a user from group U_0 (U_1), we allocate a rating to an item in category I_0 (I_1) with probability β and an item in category I_1 (I_0) with probability $1 - \beta$. The goal is to study the effect of the data bias in the explanations, so we vary the parameter β to take values in $\{0.5, 0.6, 0.7, 0.8, 0.9\}$. We also initialized items with $k = 5$ ratings. To preserve the bias in ratings initialization, the items in I_0 were rated by users in U_0 with the probability β and by users in U_1 with the probability $1 - \beta$ (similarly for the items in I_1).

We also want to investigate the effect of item popularity, so we varied the probability distribution with which we select to assign an item to a user with an item category. We generated popularity distributions for the items utilizing Zipf's Law. For the Zipf law parameter a we used parameters $a \in \{1, 1.1, 1.3, 1.5, 1.7\}$, where higher parameter value, implies more skewed distribution, while value $a = 1$ results



(a) Synthetic Data Explanations



(b) Greedy vs Sort

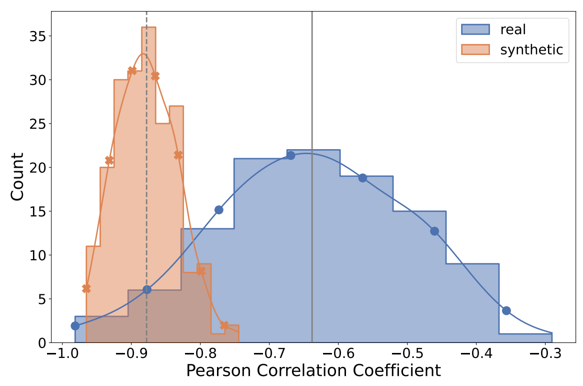
Figure 4.2: Individual item explanations

in a uniform distribution.

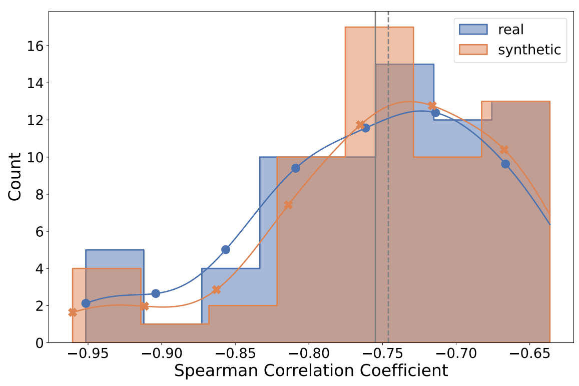
4.2 Explanations to individual users

We first experiment with explanations for individual users. In the Movies dataset, we select the target group I to be the movies in the Romance category (group R). We consider users with initial $B(R|u) < 1$, and we seek explanations that will result in $B(R|u) \geq 1$ (target $\theta = 1$). We first consider the Greedy algorithm. To study the how the initial bias affects the complexity of our explanations, we sampled 20 users in three different ranges of initial $B(R|u)$ values: $(0.65, 0.75)$, $(0.75, 0.85)$, and $(0.75, 0.85)$. Figure 4.1a plots the size of the explanation E_u and the resulting (average) $B_{E_u}(R|u)$. We observe that the more biased the nodes initially towards Action, the larger the complexity of the explanation. However, even for large Action bias, we can explain it with a small number of edges (no more than 12 for $B(R|u) \approx 0.7$ and less than 5 for $B(R|u) \approx 0.9$). An example of the explanation movies is shown in Figure 4.1b. The selected movies are all known Action movies.

We performed some additional measurements in order to better understand the type of movies that the algorithm selects as explanations, and more specifically how the popularity of the movie affects the selection. For a user u, we compute the correlation between the Δ-values, $\Delta(u, R, (u, i))$, of the removed edges and their degree popularity. The resulted histogram of the correlation coefficients (both value and rank correlation) for a sample of 200 users appears in Figure 4.3 . We observe a clear neg-



(a) Δ-values



(b) Explanation rank

Figure 4.3: Individual user correlations with item popularity

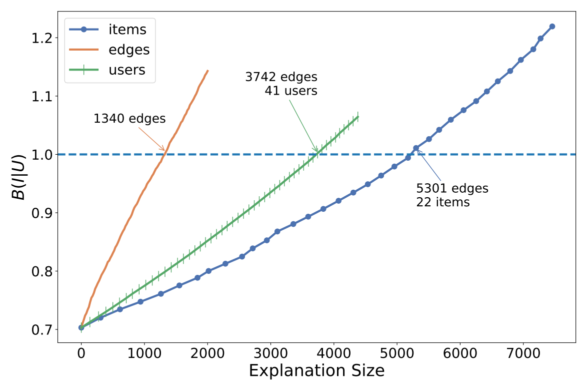
ative correlation (average Pearson correlation -0.64, average Rank Correlation -0.76), meaning that the movies selected have small degree. It is thus the case that removing links to fringe movies has a stronger effect than removing links to popular movies. This can also be deduced from Equation 3.4, where the target of the edge selected must have lower probability of reaching the target group, than the other neighbors of the user. Unpopular Action movies are less likely to lead Romance category.

To further understand the explanations produced we perform experiments with synthetic data. We vary the bias in the data β, as well as the skewness of the degree distribution a. We observe that as the bias increases it the complexity of the explanations increases. Increasing the skewness of the degree distribution also increases the complexity, since it has an effect on the resulting bias. We also compute the correlation coefficient for the synthetic data as well, and we observe an even stronger negative correlation between the Δ-value of an edge and the popularity of the movie.

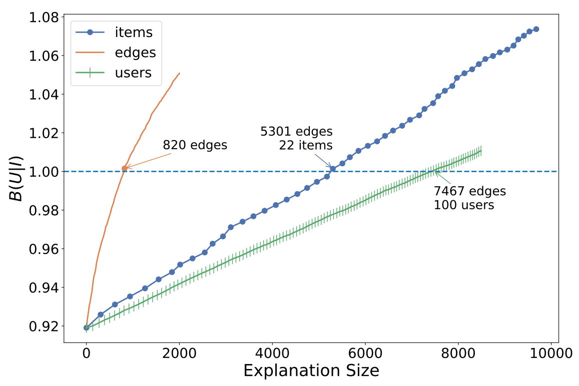
Finally, we compare the Greedy algorithm with the efficient heuristic that simply sorts the edges by the Δ values. We observe that we gain considerably in efficiency, with only a small increase in the explanation complexity for large values of target bias θ. Thus, the sorting algorithm is a viable alternative to the Greedy solution.

4.3 User-group explanations

We now consider explanations to user groups. In the Movies dataset we set the target item group to be the Romance category R, and we consider the user group consisting

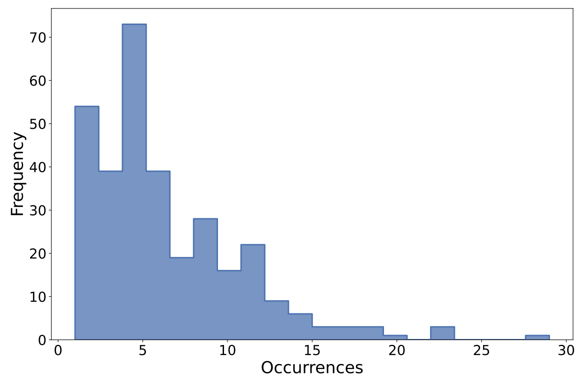


(a) User Group Explanations

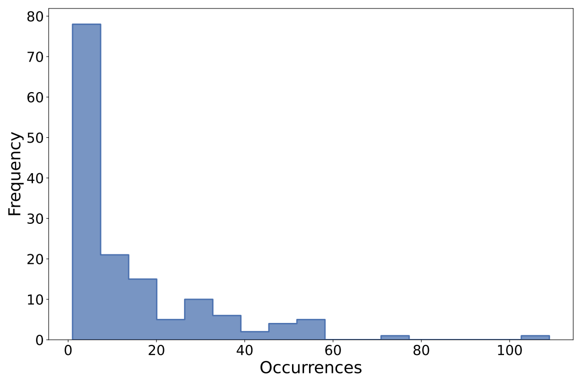


(b) Item Group Explanations

Figure 4.4: User-group and Item-group explanations



(a) Users



(b) Items

Figure 4.5: User Group EdgeExplain Occurrences

of male users M. We have that $B(R|M) \approx 0.70$, so there is a bias of the recommender against the Romance category when producing scores for the Male users, which we want to explain.

We consider the three different algorithms for producing explanations that we described in Section 3.4: EdgeExplain, UserExplain, ItemExplain. Each algorithm produces a different type of explanation. We plot them together in Figure 4.4b, where the x-axis shows the number of edges removed and the y-axis $B(R|M)$. On the plot we also show the number of users selected for UserExplain, and the number of items "removed" for ItemExplain, to achieve the target bias value.

We observe that in terms of edges removed the EdgeExplain algorithm has the most efficient explanation, followed by the UserExplain algorithm, and then the ItemExplain algorithm. This is expected since the last two produce a different type of explanation. These explanations are interesting in their own right. The UserExplain algorithm can explain the bias by affecting only a small subset of 44 users, while the ItemExplain algorithm produces an explanation with just 22 movies. Looking at the selected edges of EdgeExplain, we observed that they involve 320 distinct users, and 148 distinct movies. The occurrence distribution for the users indicates that we never remove a lot of edges from a user. On the other hand, the occurrence distribution for movies is more skewed, with many movies appearing a few times, and a few movies having several edges removed. The occurrence histograms appear in Figure 4.5.

We also explored the characteristics of the explanations. The EdgeExplain algorithm tends to select users with low degree and few edges towards Action (negative correlation under -0.6). Removing a single edge from such users has a strong effect, as it transfers significant amount of probability to the Romance group. This is contrast to the UserExplain algorithm who selects users with high degree, and several ratings in Action (correlation 0.91). Removing all of these edges results in high increase of $B(R|M)$. Interestingly, there is zero overlap between the users affected by the two algorithms.

The ItemExplain algorithm tends to select movies that are overall popular (correlation 0.86). The 22 movies selected by ItemExplain appear also in the top-100 edges selected by EdgeExplain (66% of the edges), while 16 out of the 22 selected movies appear in the top-100. The top-5 selections of the ItemExplain algorithm are: “Air Force One”, “The Godfather”, “The Princess Bride”, “Independence Day”, “Star Treck: First Contact”. We see that the list contains popular movies that are also popular outside the Action genre, such as “The Godfather” or “The Princess Bride”.

We also experimented with synthetic data. Results are shown in Figure 4.6. We observe again that as the bias increases the explanation complexity also increases, while increasing the skewness of the degree distribution (high a values) results in higher explanation size.

4.4 Individual item explanations

We now consider explanations to individual items. In the Movies dataset we set the target user group U to be the group of male users M. We consider items with initial $B(M|i) < 1$, and we seek explanations that will result in $B(M|i) \geq 1$ (target $\theta = 1$). We

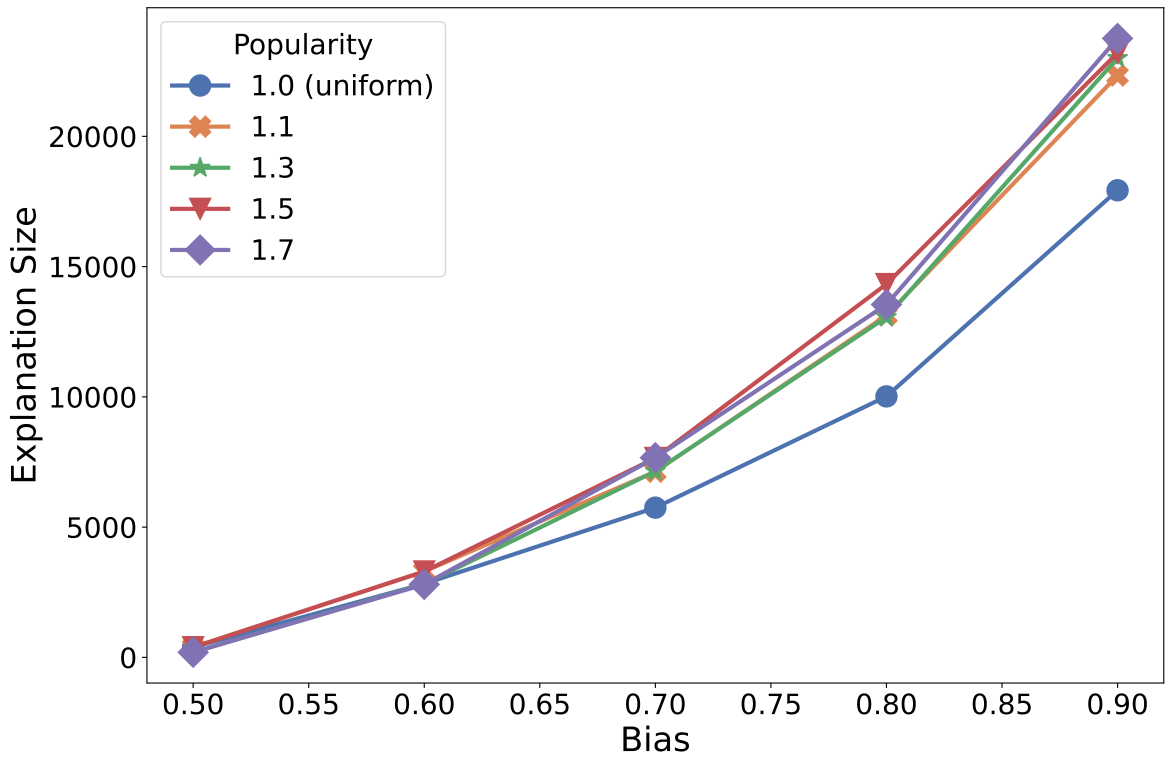
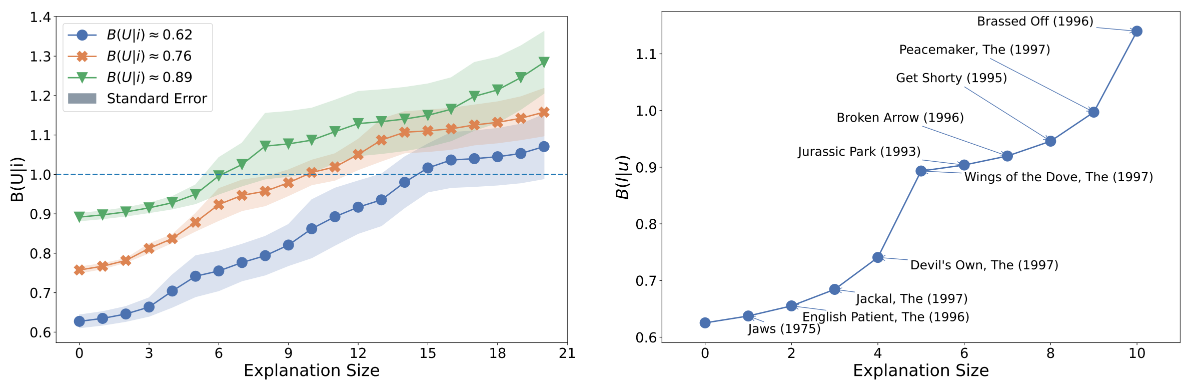


Figure 4.6: User-group explanations for synthetic data

create again samples of 20 items, for three different ranges of initial $B(M|i)$ values: $(0.55, 0.65)$, $(0.7, 0.8)$, and $(0.85, 0.95)$.

Figure 4.7a plots the size of the explanation E_i and the resulting (average) $B_{E_i}(M|i)$. We observe that the higher the bias against Males (lower $B(M|i)$), the the larger the complexity of the explanation. However, we can still explain the bias with a small number of edges (15 on average for $B(M|i) \approx 0.6$ and 6 on average for $B(R|u) \approx 0.9$). An example of the explanation movies is shown in Figure 4.7b. In the selected movies there are known Action movies like "Jaws" or "The Jackal". However, there are also Romance movies like "English Patient" or "The Wings of the Dove". This choice is made because the algorithm removes edges from users that have seen the object movie and also with low number of seen movies in general. This way we have a greater effect on the score allocated from random walk to "Oscar & Lucinda" removing a Romance movie instead of removing an Action from a user with more ratings. Thus, the algorithm prioritize users with low number of ratings.



(a) Explanation size (b) Oscar & Lucinda explanations

Figure 4.7: Individual item Explanations

4.5 Item-group explanations

We now consider explanations to item groups. We set again the target user group U to be the group of male users M, and the target item group to be the Romance category R. We have that $B(M|R) \approx 0.92$, so there is a bias of the recommender towards Males when producing scores for the Romance, which we want to explain.

We consider again the three different algorithms for producing explanations for groups: EdgeExplain, UserExplain, ItemExplain. Note that the selections of the algorithms are exactly the same as for the user-group case. The results in the bias ratio are different, as it is shown in Figure 4.4.

We first observe that the EdgeExplain achieves the target value much faster, and with a steep increase. The ItemExplain algorithm coincidentaly achieves the target bias at the same number of movies as for the user-group case. The algorithm in this case is better than the UserExplain algorithm, which performs much worse, both in terms of number of edges removed and number of users affected. The UserExplain algorithm selects users with many edges to the Action category. This increases $R(M, R)$, but it also increases $R(F, R)$, so the increase in $B(M|R)$ is small. This is in contrast with the EdgeExplain algorithm that selects edges from users with small degree, that cause large increase to $R(M, R)$ but small increase to $R(F, R)$.

Chapter 5

Related Work

The problem of fairness in recommendations has received a lot of recent attention [1, 2]. Several formulations of the problem have been proposed looking at different levels, sides, and perspectives of fairness. A general distinction is on whether fairness is considered at the level of individuals or groups [11]. In recommendations in particular, there are also many sides involved [3]. Consumer, or user fairness looks at the users receiving the recommendations, whereas producer or item fairness looks at the items being recommended. In this work, we provide explanations both at the individual and group level as well as both from the user and the item side.

Various perspectives of fairness have also been captured. One perspective is that the prediction errors of the item ratings must be similar across groups or individuals [12, 13]. Accuracy based fairness is also formulated using pairwise metrics [14]. Another perspective is fair exposure, for example, allocating exposure to items in recommendation lists proportional to their relevance [15, 16]. Finally, calibration asks that the predicted proportions of the various recommended items, or groups agree with the actual proportions of the items, or groups in the user preferences [17, 18]. In this work, we offer a general method for explaining unfair behavior and applied it to explain cases where there is discrepancies between predictions and group parity.

Explainability in AI is getting increasing attention. It is achieved by using interpretable and transparent models, or by generating post-hoc explanations for opaque models. A common approach in the latter case are attribution-based methods including methods that quantify how much the output is changed when an input variable is perturbed and methods that quantify marginal effects of variables on the output compared to a reference model [19]. Well-known examples of such methods are LIME [20] and DeepLIFT [21]. As opposed to attribution techniques, counterfactual explanations produce small changes in the input so that a different prediction is made [5]. In this work, we take a post-hoc countefactual-based approach.

There has been much work on explaining recommendation results as well [4]. Here we focus on counterfactual-based explanations for explaining unfairness. Counterfactual explanations for recommendations explore either item features, or user actions. An example of the first approach is CountER that formulates an optimization problem to generate minimal changes on the features of an item such that the recommendation decision about the item is reversed [6]. Our approach falls in the latter case where a counterfactual explanation is a set of user actions that, when removed, the recommendations change. Prince [7] follows this approach for graph recommenders and looks for a set of minimal user actions that, if removed, the top recommendation item will be replaced by a different item. ACCENT extends the user action approach to neural recommenders [8]. Here, we extend the user action approach for explaining unfairness. The only other work on counterfactual explanations for unfairness in recommendations that we are aware of is [22] that follows an item feature approach.

Finally, we propose modification of the interaction graph. There has also been a line work on graph perturbations to achieve specific properties. For example, previous research has studied the importance of edges in a network such that link recommendation algorithms result in increasing the Pagerank of underrepresented groups [23], while rewiring edges was proposed to decrease paths to polarized content [24].

Chapter 6

Conclusions

In this work we considered the problem of defining counterfactual explanations for bias of recommendation algorithms. We considered different types of bias, and provided definitions for the explanations for these biases. We the case of a random walk recommender, and we provided efficient algorithms for computing different types of explanations. We validated our approach with experiments on real and synthetic data.

For future work, we are interested in extending our approach to more recommendation algorithms, and other graph-based approaches, such as GNNs. We also plan to consider other definitions of fairness and explanations for these definitions.

Bibliography

[1] E. Pitoura, K. Stefanidis, and G. Koutrika, "Fairness in rankings and recommendations: an overview," *VLDB J.*, vol. 31, no. 3, pp. 431–458, 2022.

[2] Y. Wang, W. Ma, M. Zhang, Y. Liu, and S. Ma, "A survey on the fairness of recommender systems," *CoRR*, vol. abs/2206.03761, 2022.

[3] R. Burke, "Multisided fairness for recommendation," *CoRR*, vol. abs/1707.00093, 2017. [Online]. Available: http://arxiv.org/abs/1707.00093

[4] Y. Zhang and X. Chen, "Explainable recommendation: A survey and new perspectives," *Found. Trends Inf. Retr.*, vol. 14, no. 1, pp. 1–101, 2020.

[5] S. Verma, J. P. Dickerson, and K. Hines, "Counterfactual explanations for machine learning: A review," *CoRR*, vol. abs/2010.10596, 2020. [Online]. Available: https://arxiv.org/abs/2010.10596

[6] J. Tan, S. Xu, Y. Ge, Y. Li, X. Chen, and Y. Zhang, "Counterfactual explainable recommendation," in *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*. ACM, 2021, pp. 1784–1793.

[7] A. Ghazimatin, O. Balalau, R. S. Roy, and G. Weikum, "PRINCE: provider-side interpretability with counterfactual explanations in recommender systems," in *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, J. Caverlee, X. B. Hu, M. Lalmas, and W. Wang, Eds. ACM, 2020, pp. 196–204.

[8] K. H. Tran, A. Ghazimatin, and R. S. Roy, "Counterfactual explanations for neural recommenders," in *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 2021, pp. 1627–1631.

[9] A. N. Nikolakopoulos and G. Karypis, "Recwalk: Nearly uncoupled random walks for top-n recommendation," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, J. S. Culpepper, A. Moffat, P. N. Bennett, and K. Lerman, Eds. ACM, 2019, pp. 150–158.

[10] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, dec 2015. [Online]. Available: https://doi.org/10.1145/2827872

[11] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. S. Zemel, "Fairness through awareness," in *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012.* ACM, 2012, pp. 214–226.

[12] S. Yao and B. Huang, "Beyond parity: Fairness objectives for collaborative filtering," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 2017, pp. 2921–2930.

[13] B. Rastegarpanah, K. P. Gummadi, and M. Crovella, "Fighting fire with fire: Using antidote data to improve polarization and fairness of recommender systems," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019.* ACM, 2019, pp. 231–239.

[14] A. Beutel, J. Chen, T. Doshi, H. Qian, L. Wei, Y. Wu, L. Heldt, Z. Zhao, L. Hong, E. H. Chi, and C. Goodrow, "Fairness in recommendation ranking through pairwise comparisons," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019.* ACM, 2019, pp. 2212–2220.

[15] A. Singh and T. Joachims, "Fairness of exposure in rankings," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018.* ACM, 2018, pp. 2219–2228.

[16] A. J. Biega, K. P. Gummadi, and G. Weikum, "Equity of attention: Amortizing individual fairness in rankings," in *The 41st International ACM SIGIR Conference*

on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018. ACM, 2018, pp. 405–414.

[17] V. Tsintzou, E. Pitoura, and P. Tsaparas, "Bias disparity in recommendation systems," in *Proceedings of the Workshop on Recommendation in Multi-stakeholder Environments co-located with the 13th ACM Conference on Recommender Systems (RecSys 2019), Copenhagen, Denmark, September 20, 2019*, ser. CEUR Workshop Proceedings, vol. 2440. CEUR-WS.org, 2019.

[18] H. Steck, "Calibrated recommendations," in *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018.* ACM, 2018, pp. 154–162.

[19] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 2017, pp. 3319–3328.

[20] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should I trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016.* ACM, 2016, pp. 1135–1144.

[21] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 2017, pp. 3145–3153.

[22] Y. Ge, J. Tan, Y. Zhu, Y. Xia, J. Luo, S. Liu, Z. Fu, S. Geng, Z. Li, and Y. Zhang, "Explainable fairness in recommendation," in *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022.* ACM, 2022, pp. 681–691.

[23] S. Tsioutsiouliklis, E. Pitoura, K. Semertzidis, and P. Tsaparas, "Link recommendations for pagerank fairness," in *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022.* ACM, 2022, pp. 3541–3551.

[24] F. Fabbri, Y. Wang, F. Bonchi, C. Castillo, and M. Mathioudakis, “Rewiring what-to-watch-next recommendations to reduce radicalization pathways,” in *WWW ’22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*. ACM, 2022, pp. 2719–2728.

[25] K. S. Miller, “On the inverse of the sum of matrices,” *Mathematics Magazine*, vol. 54, no. 2, pp. 67–72, 1981.

[26] P. G. Doyle and J. L. Snell, *Random Walks and Electric Networks*, ser. Carus Mathematical Monographs. Washington, DC: Mathematical Association of America, 1984, no. Book 22.

[27] C. M. Grinstead and J. L. Snell, *Introduction to Probability*. AMS, 2003. [Online]. Available: http://www.dartmouth.edu/~chance/teaching_aids/books_articles/probability_book/book.html

Appendix A

Appendix A

A.1 Estimating the effect of edge removals from a user

We will provide a proof for Equations 3.5 and 3.6. The proof also applies to the case of a single edge removal, where the set of edges E_x removed contains a single edge ($|E_x| = 1$).

Proof. Let P and P' denote the transition matrices of the recommender random walk on the graphs G and G' before and after the removal of the edge of edges E_x respectively. The set E_x contains (directed) edges emanating from node x, that is, it is of the form $E_x = \{(x, y) : y \in Y\}$, for a subset Y of the neighbors of x. Abusing the notation, since the source of the edges is fixed to x, we will use E_x to denote both the set of edges and the set of neighbors Y of x from which we remove the edges. We assume that $|E_x| = k$, and that $d_x > k$.

To prove our theorem, we first write the transition matrix P' as the sum of the transition matrix P and a rank-1, perturbation matrix B, that is, $P' = P + B$. We use B_i to denote the i-th row of matrix B, and we have:

$$B_i = \begin{cases} 0, & i \neq x \\ \frac{k}{d_x - k} P_x - \frac{1}{d_x - k} \mathbf{e}_{E_x}^T, & i = x \end{cases}$$

where $\mathbf{e}_{E_x}$ is the vector with 1 at the positions of nodes in E_x, and zero everywhere else.

We want to estimate

$$Q' = \gamma \left(I - (1-\gamma)P'\right)^{-1} = \gamma \left(I - (1-\gamma)(P+B)\right)^{-1}.$$

To do so, we exploit a lemma [25] that states that for a non-singular matrix M and a rank-1 matrix H, such that $M+H$ is nonsingular, we have:

$$(M+H)^{-1} = M^{-1} - \frac{1}{1+g} M^{-1} H M^{-1},\ g = tr(HM^{-1})$$

Applying for $M = (I - (1-\gamma)\ P)$ and $H = -(1-\gamma)\ B$, and using the fact that $Q = \gamma M^{-1}$:

$$\begin{aligned} Q' &= \gamma (M+H)^{-1} \\ &= \gamma M^{-1} - \gamma \frac{1}{1+g} M^{-1} H M^{-1},\ g = tr(HM^{-1}) \\ &= \gamma \frac{Q}{\gamma} - \gamma \frac{1}{1+h}\ \frac{Q}{\gamma} \left(-(1-\gamma)B\right) \frac{Q}{\gamma},\ h = tr\left(-(1-\gamma)B \frac{1}{\gamma} Q\right) \\ &= Q + \frac{\frac{(1-\gamma)}{\gamma}}{1 - \frac{(1-\gamma)}{\gamma} q} Q\ B\ Q,\ \text{where}\ \ q = tr(B\ Q) \end{aligned} \tag{A.1}$$

With mathematical manipulations, we get:

$$\begin{aligned} BQ_{ui} &= \begin{cases} 0, & i \neq x \\ \frac{k}{d_x - k}\ \left(\frac{1}{d_x} \sum_{j \in D_x} Q_{ji} - \frac{1}{k} \sum_{j \in E_x} Q_{ji} \right), & i = x \end{cases} \\ QBQ_{ui} &= \frac{k}{d_x - k} Q_{ux}\ \left(\frac{1}{d_x} \sum_{j \in D_x} Q_{ji} - \frac{1}{k} \sum_{j \in E_x} Q_{ji} \right) \end{aligned}$$

Substituting in Equation A.1, and using the fact that $q = tr(B\ Q) = BQ_{xx}$ we have:

$$Q'_{ui} = Q_{ui} + Q_{ux} \frac{\frac{(1-\gamma)}{\gamma} \left(\frac{1}{d_x} \sum_{j \in D_x} Q_{ji} - \frac{1}{k} \sum_{j \in E_x} Q_{ji} \right)}{\frac{d_x - k}{k} - \frac{(1-\gamma)}{\gamma} \left(\frac{1}{d_x} \sum_{j \in D_x} Q_{jx} - \frac{1}{k} \sum_{j \in E_x} Q_{jx} \right)} \tag{A.2}$$

We know that for any nodes k, ℓ, $\mathbf{p}_k(\ell) = Q_{k\ell}$. Also, $d_x = |D_x|$ and $|E_x| = k$. Therefore:

$$\mathbf{p}_u(i|E_x) = \mathbf{p}_u(i) + \mathbf{p}_u(x) \frac{\frac{(1-\gamma)}{\gamma} \left(\frac{1}{|D_x|} \sum_{j \in D_x} \mathbf{p}_j(i) - \frac{1}{|E_x|} \sum_{j \in E_x} \mathbf{p}_j(i) \right)}{\frac{|E_x| - |D_x|}{|D_x|} - \frac{(1-\gamma)}{\gamma} \left(\frac{1}{|D_x|} \sum_{j \in D_x} \mathbf{p}_j(x) - \frac{1}{|E_x|} \sum_{j \in E_x} \mathbf{p}_j(x) \right)}$$

□

A.2 Computing personalized random walk probabilities using absorbing random walk

We will now show that the absorption probabilities $\alpha_x(A_Z)$ of the absorbing random walk we defined in Section 3.3 compute the personalized random walk probabilities $\mathbf{p}_x(Z)$.

The absorbing random walk is defined by the transition matrix R:

$$R = \begin{bmatrix} T & B \\ 0 & I_4 \end{bmatrix}$$

The matrix T is the transition matrix between transient nodes (non-absorbing). In our case, we have $T = (1-\gamma)P$, where P the transition probability matrix of the random walk algorithm, and γ the jump probability. Matrix B has the transition probabilities from the transient nodes to the absorbing nodes. In our case, we have four absorbing nodes, so it is an $n \times 4$ matrix, where each column corresponds to an absorbing nodes A_Z. For the column B_Z we have $B_Z(i) = \gamma$ if $i \in Z$, and 0 otherwise.

Using existing theory on absorbing random walks [26, 27]. to compute the absorption probabilities, we compute the fundamental matrix $F = (I - (1-\gamma)P)^{-}1$, and then we have that $\alpha_x(A_Z) = FB_Z(x)$. Note that $F = \frac{1}{\gamma}Q$. Therefore, we have:

$$\begin{aligned} FB_Z(x) &= \frac{1}{\gamma}QB_Z(x) \\ &= \frac{1}{\gamma}Q_xB_Z \\ &= \frac{1}{\gamma}\mathbf{p}_xB_Z \\ &= \sum_{i \in Z}\frac{1}{\gamma}\mathbf{p}_x(i)\gamma \\ &= \sum_{i \in Z}\mathbf{p}_x(i) \\ &= \mathbf{p}_x(Z) \end{aligned}$$

In the derivation above we used the fact that the x-th row Q_x of matrix Q stores the personalized vector $\mathbf{p}_x$, and the fact that for a subset of nodes Z (e.g., $Z = I$), $\mathbf{p}_x(Z) = \sum_{i \in Z}\mathbf{p}_x(i)$.

Short Biography

Leonidas Zafeiriou was born in Thessaloniki in 1997. He graduated from Experimental School of University of Thessaloniki in 2015. He recieved his diploma in Engineering in 2021 from the Department of Computer Science & Engineering in the University of Ioannina. His diploma thesis has title ”Image Keyword Spotting with Quaternion-analysis based Methods” under the supervision of Professor Christophoros Nikou and the guidance of Assistant Professor Giorgos Sfikas. He continued his studies as an MSc student at the same department, working under the supervision of Assoc. Professor Panayiotis Tsaparas and the guidance of Professor Evaggelia Pitoura.