1-2022

Beyond accuracy in machine learning.

Aneseh Alvanpour
University of Louisville

BEYOND ACCURACY IN MACHINE LEARNING

By

Aneseh Alvanpour
M.Sc., Computer Engineering and Computer Science,
University of Louisville, Louisville, KY

A Dissertation
Submitted to the Faculty of the
J.B. Speed School of Engineering of the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy in Computer Science and Engineering

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, Kentucky

May 2022

BEYOND ACCURACY IN MACHINE LEARNING

By

Aneseh Alvanpour
M.Sc., Computer Engineering and Computer Science,
University of Louisville, Louisville, KY

A Dissertation Approved On

May 12, 2022

by the following Dissertation Committee:

Dr. Olfa Nasraoui, Dissertation Director

Dr. Dan Popa

Dr. Hichem Frigui

Dr. Nihat Altiparmak

Dr. Hui Zhang

ACKNOWLEDGEMENTS

First and foremost, I want to thank my advisor Dr. Olfa Nasraoui for all her valuable guidance and support. This dissertation would not have been possible without her time, ideas and encouragement. Also, I would like to thank the members of my committee Dr. Dan Popa, Dr. Hichem Frigui, Dr. Nihat Altiparmak, and Dr. Hui Zhang, for their feedback on my dissertation and for their kind patience.

I would also like to thank my family and friends for their love and attentions. Special thanks to my parents, Moones and Cyruse, my lovely sister and brother-in-law, Anahita and Kamran; and my niece and little princess Artemis, as well as my beloved brothers, Arash and Arman; all of whom have given me their unconditional support, love and encouragement throughout my years pursuing my education and research goals. I will be forever grateful for their love.

Last but not least, a special thanks to my friends and fellow lab members for their friendship and precious collaboration.

ABSTRACT

BEYOND ACCURACY IN MACHINE LEARNING

Aneseh Alvanpour

May 12, 2022

Machine Learning (ML) algorithms are widely used in our daily lives. The need to increase the accuracy of ML models has led to building increasingly powerful and complex algorithms known as black-box models which do not provide any explanations about the reasons behind their output. On the other hand, there are white-box ML models which are inherently interpretable while having lower accuracy compared to black-box models.

To have a productive and practical algorithmic decision system, precise predictions may not be sufficient. The system may need to have transparency and be able to provide explanations, especially in applications with safety-critical context such as medicine, aerospace, robotics, and self-driving vehicles; or in socially-sensitive domains such as credit scoring and predictive policing. This is because having transparency can help explain why a certain decision was made and this in turn could be useful in discovering possible biases that lead to discrimination against any individual or group of people.

Fairness and bias are another aspect that needs to be considered in evaluating ML models. Therefore, depending on the application domain, *accuracy*, *explainability* and *fairness* from bias may be necessary in building a practical and effective algorithmic decision system. However, in practice, it is challenging to have a model that optimizes all of these three aspects simultaneously.

In this work, we study ML criteria that go beyond accuracy in two different problems:

1) in collaborative filtering recommendation, where we study explainability and bias in addition to accuracy; and 2) in robotic grasp failure prediction, where we study explainability in addition to prediction accuracy.

We discuss the trade-offs between accuracy, explainability and bias and we study the effect of popularity debiasing on *Explainable Matrix Factorization (EMF)* by applying an *inverse propensity scoring*. To overcome the popularity bias problem in recommendation, we propose new algorithms to recommend more explainable and less popular items to users. Our experiments compare the performance of the proposed algorithms to existing methods in terms of *accuracy*, *explainability*, *novelty*, and *diversity* on publicly available benchmark data and study the trade-offs between these three aspects.

Moreover, we report on a study of explainable robotic grasp failure prediction, comparing classical interpretable white-box ML models, more sophisticated black-box models, and an intermediate, glass-box model, based on the explainable boosting machine (*EBM*). The results show that applying *EBM* leads to a gain in performance, not only in accuracy metrics but also in implementation time, without sacrificing interpretability. While the Logistic Regression white-box model (LR) is faster at prediction time than the glass-box model (*EBM*), LR is only globally interpretable and is not locally explainable. To obtain a local explanation, it takes almost 2 orders of magnitude longer for LR (which relies on LIME for the explanation generation) compared to a local explanation generated by *EBM* for its own prediction. Our experiments further show that applying proper effort to the object and particularly the effort of joint 1 has important role in the stability of the grasps. Thus our results match with the mechanical concepts.

In addition, we extend our study of explainable robotic grasp failure prediction by evaluating the consistency between the explanations generated from different post-hoc explanation methods, in a third case study. The results show that the local explanations generated from the output of a black-box model (Random Forest, in our case) are more consistent in ranking of feature contributions than the local explanations generated from the output of a white-box model (a Decision Tree classifier in our study). This shows

that the consistency of the explanations (rankings of important feature contributions in our case), generated from different explanation methods, depends on the type of predictive ML model used to make the predictions in the first place.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION AND MOTIVATION

1.1 Motivation and Problem statement

1.1.1 Motivation

Machine Learning (ML) algorithms are being increasingly used in many sectors that affect society at large. The need to increase the accuracy in performance of ML models, led to building more powerful and complex algorithms known as black-box models which do not provide explanations about the reasons behind the output of the models. On the other hand, there are white-box models which are inherently interpretable while generally having lower accuracy compared to black-box models.

To have a productive and practical algorithmic decision system, precise predictions may not be sufficient. The system needs to have transparency such as by being able to provide explanations, especially in applications with safety-critical contexts such as medicine, aerospace, robotics, and self-driving vehicles, or in socially-sensitive domains such as credit scoring and predictive policing. Having transparency can help explain why a certain decision was made and it can be useful in discovering possible biases leading to discrimination against any individual or group of people.

Fairness and bias are another dimension that needs to be considered in evaluating ML models. Therefore, depending on the application, *accuracy*, *explainability* and *fairness* may be necessary in building a practical and effective algorithmic decision system. However, in practice, it is challenging to have a model that optimizes all of these three aspects simultaneously.

Recommender Systems provide personalized suggestions for consumers to help them discover their preferred items among millions of songs, movies, news, jobs, courses and

friends. They also support providers to gain more benefits while being used in many applications such as e-commerce and social media.

Focusing on achieving higher accuracy by discovering the most relevant items to users, led to using black-box algorithms such as Matrix Factorization (MF) [3] which provides highly accurate predictions about users' interests, but no explanations about the predictions of the model. A recent work by Abdollahi and Nasraoui [4, 5] introduced an explainability constrained MF technique called *Explainable Matrix Factorization* (EMF). By adding soft explainability constraints to the MF objective function, it brought users closer to their explainable items in latent space and showed significant improvement in explainability and accuracy in recommendations.

Analysing the sensitivity of EMF to explainability and accuracy, [4, 5] found that there is a positive correlation between explainability and accuracy. They showed that by increasing the explainability parameter (λ), there is no sacrifice in the accuracy of EMF. However due to the inherent bias of the input rating data, the explainability scores which aggregate the neighbors' ratings may be affected by popularity bias. So far only one recent study addressed both explainability and debiasing together [6]; however it focuses on implicit feedback data, which is outside the scope of this work. In contrast, this work focuses on explicit (rating) data.

To mitigate the effect of popular items on *EMF's* explainability and accuracy, we propose a *Debiased Explainable Matrix Factorization* (Debiased EMF) method which uses an *inverse popularity propensity score* (IPS) in its objective function.

Grasping reliability is important in many robotic tasks involving human safety or material costs. For this reason, predicting grasp failures before they occur can give timely warnings about potential reliability risks during manipulation. These predictions can guide decision making by both humans interacting with the robot and by design engineers seeking to improve the robot performance.

Despite advances in robot control, imprecision in sensing and actuation is still a challenge in making a robot's grasp more stable [7]. This has motivated using ML to

predict failures before they occur. However, Black-Box models fail to explain the reasons for predicted failures, and thus give no clues about why failures are predicted to occur, whether to trust the prediction of failure, nor how to avoid failures for instance by different designs. This is why explainability in ML could provide a solution for the robotic grasp failure prediction challenge.

1.1.2 Problem Statement and Scope

We study ML criteria that go beyond accuracy in two different domains: 1) in collaborative filtering recommendation where we study explainability and bias in addition to accuracy, and 2) in robotic grasp failure prediction where we study explainability in addition to prediction accuracy.

To achieve a balance between *accuracy*, *explainability* and *bias* in matrix factorization based recommendation systems, we propose a new algorithm which provides both Debiasing and Explainability simultaneously, called *Debiased Explainable Matrix Factorization* (Figure 1.1). Moreover, we compare the effect of applying the *Inverse Propensity Weighting or Scoring (IPW or IPS)* in the following two different terms of the combined loss function:

1. The rating prediction loss

2. The explainability regularization term

We would like to evaluate the effectiveness of down-weighting popular items in rating prediction for unseen items and in recommending more explainable items to users. We limit the scope of our work to Collaborative Filtering, since this paradigm is considered the state of the art and drives the most flexible recommendation engines in a wide variety of domain.

In real world recommendation systems, popularity bias can happen due two other biases; *interaction bias*, when users tend to have more interaction with popular items, and *presentation bias*, when recommender systems show more popular items compared to items in the long-tail (less popular items) [8]. Since the dataset that we use for our study does

	Debiased	Not Debiased
Explainable	**Debiased_EMF_1** **Debiased_EMF_2** **Debiased_EMF_1_2**	EMF [1,2]
Unexplainable	Populaity_Propensity_MF [54]	MF [10]

Figure 1.1: Explainability and bias trade-offs in the proposed debiasing recommendation methods (in bold) and existing competitive baselines.

not provide any information to distinguish between interaction bias and presentation bias, we assume that popularity bias comes from the long-tail problem and presentation of more popular items in the recommendation list to users.

1.2 Research Contributions

In this work, we study ML criteria that go beyond accuracy in two different problems where accuracy alone may not be sufficient as a performance goal: 1) in collaborative filtering recommendation where we study explainability and bias in addition to accuracy and 2) in robotic grasp failure prediction where we study explainability in addition to prediction accuracy.

1. We propose new recommendation algorithms which provide both Debiasing and Explainability simultaneously, called *Debiased Explainable Matrix Factorization* to study the trade-offs between *accuracy*, *explainability* and *bias* (Fig. 1.1).

2. We study the effectiveness of down-weighting popular items in rating prediction for unseen items and in recommending more explainable items to users.

3. We explore the trade-offs between prediction accuracy and explainability in another ML application, namely robotic grasp failure prediction by comparing the performance of classical white-box models and different sophisticated black-box models as well as an intermediate model called glass-box model based on the explainable boosting machine (EBM) [9–11].

4. We analyze the consistency of Post-hoc Explanation methods in ranking feature contributions and finding the most responsible features for grasp failures.

1.3 Document Organization

The rest of the document is organized as follows:

- Chapter 2 reviews some background about explainability and bias in recommender systems. It also includes background about applying machine learning models from two perspectives, namely accuracy and interpretability of the models for predicting the failure of the stability of robotic grasps.

- Chapter 3 presents our proposed research and experimental results to mitigate the effect of popularity debiasing on explainability and accuracy of *Explainable Matrix Factorization (EMF)* model. Then we present the results for the proposed Debiased EMF based methods, comparing them with the baseline methods, and evaluating their performance from different perspectives; *accuracy*, *explainability*, *novelty* and *diversity*.

- Chapter 4 presents our methodologies and experimental results through three case studies on the possibility to generate explanations for predicted robotic failures, while exploring different ML modeling and explanation generation techniques that vary in their accuracy and explanation capability.

- Finally, Chapter 5 presents our conclusion.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

This chapter reviews the background related to explainability and bias in recommender systems. It also provides background about applying machine learning models from two perspectives; interpretability of the models and prediction accuracy for the stability of robotic grasps. In addition, it reviews the metrics to evaluate (measure) the similarity between global explanations generated by ML models and the local explanations by Post-hoc explanation methods.

2.1 Recommender Systems

Recommender systems are methods which suggest items that are more likely to be used by users [12, 13] . Their suggestions help users in buying items, taking courses, or choosing books to read.

To provide recommendations, recommender systems use three types of data: items, users and interactions, which explain the relation between users and items.

Items are the objects that a recommender system suggests to its users and may be characterized by their complexity and their value or utility [14]. Items such as CDs, books, movies can be considered as low complexity and value recommendations, while insurance policies, financial investments, travels, and jobs are the most complex [14]. Also, an item may be represented by a single id code or a set of features. For example, in a movie recommender system, a movie can be described by various attributes such as genre (crime, romantic, etc.), the director, and actors.

Moreover, recommender systems, based on their built-in technology, can use different features and information of the users in personalizing the recommendations. For example,

a list of ratings which each user has provided for watching movies, models the users in a collaborative-filtering technique. In demographic recommender systems, users are described by sociodemographic attributes such as age, gender, profession, and education.

2.1.1 Recommendation Techniques

Based on the domain, knowledge or the algorithm, we can classify the recommender systems techniques into different categories such as Content-based, Collaborative-filtering (CF), Demographic, Knowledge-based, Community-based, and Hybrid recommender systems. In this work we focus on CF because it is the leading modern approach in recommendation systems [15].

2.1.1.1 Collaborative Filtering

Collaborative Filtering (CF) is a popular technique and is currently the leading approach in recommender systems which uses the known preferences of a group of users to make recommendations or predictions of the unknown preferences for other users [15]. The fundamental assumption is that if users A and B rate k items similarly, they share similar tastes, and hence will rate other items similarly. Approaches differ in how they define a "rating," how they define k, and how they define "similarly" [15]. The need of finding related items and users in CF systems, has led to two main techniques of CF: the neighborhood approach and latent factor models. Neighborhood methods focus on relationships between items or users. An item-item approach models the preference of a user to an item based on ratings of similar items by the same user. Latent factor models, such as matrix factorization, transforms both items and users to the same latent factor space. The latent space tries to explain ratings by characterizing both products and users on factors automatically inferred from user feedback [14].

2.1.2 Transparency in Recommender Systems

Transparency lets the users know how the system works [14]. Most modern recommender systems rely on sophisticated black-box machine learning models that are unable to explain their output predictions. Thus, recent research has developed methods to address lack of transparency by introducing Explainable models.

Based on algorithms used in building recommender systems, explanations can be provided in different styles, such as Collaborative-based, Content-based, Demographic-based, Case-based reasoning, and Knowledge and Utility-based.

Here, we focus on collaborative-based approach which provides explanations by using user *u's* ratings on items i. Then these ratings are used to identify users that are similar in ratings to u. These similar users are often called "neighbors" as nearest-neighbors approaches are commonly used to compute similarity. Then, a prediction for the recommended item is extrapolated from the neighbors' ratings of i.

In our research, we study the effect of *inverse popularity propensity* based debiasing on *explainability*, *accuracy*, *novelty* and *diversity* on the state-of-the-art *Explainable Matrix Factorization* (EMF) method proposed by [4, 5]. The authors introduced an explainability constrained MF technique that computes the top-n recommendation list from items that are explainable. They formulated the explainability based on the rating distribution within the user's or item's neighborhood. They extended MF by adding soft explainability constraints to the objective function to bring users closer to their explainable items in latent space. Their results showed significant improvement in explainability and accuracy in recommendations. We will discuss more details about EMF in Chapter 3.

2.1.3 Bias in Recommender Systems

Recommender Systems are widely used not only in providing personalized suggestions for consumers but also in supporting providers to gain more benefits while being used in many applications such as e-commerce and social media. From the user's perspective, the recommender systems help us in discovering our preferred items among millions of songs,

movies, news, jobs, courses and friends. Although the recommender systems have important roles in our online and digital daily life, the issue of bias may affect their effectiveness.

The authors in [16] mention three reasons for having bias in the recommender systems. One reason is that user behavior data is observational rather than experimental. It means that users behave based on the items which were exposed to them. The presentation of the items in the data is another reason to cause biases in the recommender systems. For example, popular items, which get more user behaviors, have more effects on what the model learns from the data. Thus the result of the recommendation systems would be biased towards them [16].

2.1.3.1 Bias in Recommendation

1. Bias in Data

   The training data, which reflects the user behaviours, can be a source of biases in recommender systems and leading to biased decisions in the system. Whether the input data to the systems has implicit or explicit feedback from the users, we can classify this type of bias into four groups: exposure bias and position bias in implicit feedback, and selection bias and conformity bias in explicit feedback.

   (a) Bias in explicit feedback data

   Having explicit feedback from users, when users give numerical ratings to items, can cause the Selection bias. As authors mention [16], the selection bias occurs because users are free in choosing items to rate. Thus the observed ratings are not a representative sample of all ratings. In other words, the missing ratings in the data are not at random (MNAR).

   Another type of bias which roots from the explicit feedback is conformity bias. Sometimes users would like to give similar ratings to the others in a group. Therefore, the rating data does not necessarily reflect the user's true preferences [16].

(b) Bias in implicit feedback data

Sometimes the feedback data which is used in recommendation is a result of natural behaviors of users. For example, it includes information about users purchases, views, and clicks. This implicit feedback which only provides a partial signal of positive, can be the source of two other biases, exposure bias and position bias [16].

Since users are exposed to a part of specific items, the unobserved interactions do not reflect the negative preference. Therefore could cause the exposure bias [16].

Another type of bias, which is more common in e-commerce recommender systems, comes from the position of items in the recommendation list and called position bias. Items with higher position in the recommendation list are more likely to be chosen by users regardless of their relevance to the user's true preference [16].

2. Bias in Model

Generalization and better prediction of the unseen data is one of the main goals in Machine Learning. This goal can be achieved by a set of assumptions made by learning algorithms. These added assumptions to the model refer to Inductive bias [16].

3. Bias and Unfairness in Results

In addition to the biases introduced in data and model, popularity bias and unfairness are two other important biases which can be found in the results.

(a) Popularity Bias

In the recommender system training data, most of user interactions belong to a small proportion of popular items which cause the long-tail issue in the recommendation and it introduces the popularity bias [16, 17]. Using data with the long-tail problem, popular items receive higher scores by the learning algorithms while unpopular items get negative scores. Therefore, popular items are

recommended even more frequently than their initial popularity presented in the dataset [17].

Disregarding the popularity bias will cause several issues:

i. Recommending popular items will ignore users preferences and will hurt the level of personalization.

ii. Suggesting popular items, which may not always have high quality, will decrease the chance of other items to be seen by the users and will make the results unfair.

iii. Increasing the chance of exposure for popular items will cause the collected data for future use to be more imbalanced and will cause the "Matthew effect" [18] issue.

Unfairness is another common bias in recommendation systems. The authors in [19] define fairness as "absence of any prejudice or favoritism towards an individual or a group based on their intrinsic or acquired traits". Therefore Unfairness happens when a system makes unfair decisions against certain individuals or groups of people [20].

Usually the unequal representation of different groups of users based on sensitive features such as age, race, gender, education level or wealth makes the training data unbalanced. Using the unbalanced data, the model learns the behavior of the majority groups better than minority groups. Therefore, the model's output would be biased towards the majority groups and against minority groups. One example, job recommenders offer fewer ads for high-paying jobs and career coach services to women compared to men [21], [22]. This can happen due to the gender imbalance in the historical data that was used in the model training process.

2.1.3.2 Debiasing Methods

Many methods have been proposed to decrease the effect of popularity bias including Regularization [23, 24], Adversarial learning [25], Causal graph [26], and Propensity Score

[8]. So far only one recent study addressed both explainability and debiasing together [6]; however it focuses on implicit feedback data, which is outside the scope of this work. In our study we will use popularity to estimate the propensity score [27] and to debias the EMF model [4, 5] on explicit (rating) data.

2.1.4 Recommender System Evaluation Metrics

2.1.4.1 Explainability Metrics

To measure the explainability of the models we adopted Mean Explainability Precision (MEP) metric from [4, 5].

$$MEP = \frac{1}{|U|}\sum_{u \in U}\frac{|\{i : i \in top-n, Expl_{u,i} > \theta\}|}{|top-n|} \tag{2.1}$$

To get better estimation of our model explainability, we use Weighted Mean Explainability Precision ($WMEP$) which includes the explainability score of the items in its evaluation.

$$WMEP = \frac{1}{|U|}\sum_{u \in U}(Expl_{u,i})\frac{|\{i : i \in top-n, Expl_{u,i} > \theta\}|}{|top-n|} \tag{2.2}$$

We also measure the explainability with True Mean Explainability Precision ($TMEP$) which considers explainable items from the true (ground truth) top-n ranked items in addition to items that belong to top-n recommendation list (from estimated ratings).

$$TMEP = \frac{1}{|U|}\sum_{u \in U}\frac{|\{i : i \in \{(top-n) \cap (True-top-n)\}, Expl_{u,i} > \theta\}|}{|top-n|} \tag{2.3}$$

$WTMEP$ considers the effect of assigning explainability weights and considering items from true top-n ranked list simultaneously in its explainability evaluation.

$$WTMEP = \frac{1}{|U|}\sum_{u \in U}(Expl_{u,i})\frac{|\{i : i \in \{(top-n) \cap (True-top-n)\}, Expl_{u,i} > \theta\}|}{|top-n|} \tag{2.4}$$

2.1.4.2 Novelty and Diversity Metrics

To measure *Novelty* of new recommended items, we use a Popularity-based Novelty metric called Expected Free Discovery (EFD) [28] which defines item novelty as the difference between an item and "what has been observed" as shown in Equation 2.5.

$$EFD = \frac{-1}{|R|}\sum_{i \in R}(1 - P_{u,i}) \tag{2.5}$$

$$P_{u,i} = \frac{N_i}{N_U} \tag{2.6}$$

Here $P_{u,i}$ is the probability that user u has observed item i and is equal to the proportion of N_i (number of users who rated item i) to N_U (total number of users).

Given a popularity threshold θ, for each user we count number of unpopular items (items which whose popularity is less than the threshold) among the top-n recommended items to that user (D_u). Then we report the average of this count on all users as shown in Equation 2.7.

$$Diversity = \frac{1}{|U|}\sum_{u \in U}|D_u| \tag{2.7}$$

Here D_u is the count of unpopular items among the top-n recommendation list.

$$D_u = \{i \in top - n : P_{u,i} < \theta\} \tag{2.8}$$

2.2 Background on Grasp Failure Prediction

ML models have been widely used to increase the stability of robotic grasping by detecting or predicting grasping failures and taking appropriate actions to correct them [29,30]. The authors in [31] used dual robot arms and applied convolutional neural networks (CNN) to approximate the grasping points of the objects and predict grasping failures. Deep neural networks and an anthropomorphic soft hand were used to improve the reliability of grasping [32]. Partial least square (PLS) was another approach that monitored faults during robot operation, online, while the faults were detected offline [33].

Among fault diagnosis methods, in data-driven (knowledge-based) techniques, a diagnosis decision is made by comparing the performance output of the system and a learned knowledge (from the performance of the system in the past) which is based on the extracted information achieved by applying artificial intelligence to a large amount of historical data [34]. This knowledge can be obtained from qualitative or quantitative methods. The qualitative techniques are mainly grouped into Signed Directed Graph [35], Expert System (ES) [36], and Fault Tree (FT) [36]. While the quantitative methods can be divided into three main Machine Learning algorithms: Supervised Learning (such as Partial Least Squares (PLS) [37], Principal Component Analysis (PCA) [38], Support Vector Machine (SVM) [39], Neural Networks (NNs), Fuzzy Logic (FL) [40], Bayesian Classifier [41]), Unsupervised Learning (such as Nearest Neighbor [42] and K-means [43, 44]), and Reinforcement Learning [45, 46]. Despite the extensive research and advancements in object manipulation, maintaining grasp stability is still challenging in robotics due to uncertainties during the grasping process (from detecting correct finger positions on the object to applying the proper force by fingers) [47]. Thus research has been conducted to propose strategies and methods to provide high-precision grasps and to predict potential faults by applying different ML algorithms including Deep Learning methods [48]. The authors in [29] proposed an effective fault detection algorithm based on neural networks to detect faults for robot manipulators. Another study by [49] was able to classify and predict grasp failure in soft robotic hands by proposing two deep learning architectures. By applying a deep neural architecture they classified successful and unsuccessful grasps. Then by proposing a second neural architecture, composed of three convolutional layers (CNN) and two Long Short Memory Networks (LSTM), they predicted the times of failure before they occurred. Another LSTM-based model was proposed by [50] to detect faults for an industrial robot manipulator and it showed better performance in predicting faults compared to other ML models including Random Forest [51], SVM, and k-NN+ DTW (k-Nearest Neighbor and Dynamic Time Warping [52]).

2.3 Interpretable Machine Learning Methods

Machine learning algorithms have been applied to help robots learn how to work and make decisions based on information received from sensors. Information could be in the form of image features from cameras or positions or velocity of gripper joints [53]. Despite advances in robot control, imprecision in sensing and actuation still challenge the stability of a robot's grasp [7].

Interpretable ML methods can be categorized based on whether interpretability is achieved by limiting the complexity of the ML model (hence called intrinsically interpretable methods) or by applying methods that analyze the learned ML model after training (hence called post-hoc interpretable methods) [54]. Algorithms that produce simple structures such as Decision Trees (DT) [55] and linear models, like Logistic Regression (LR) [56], are intrinsically interpretable. This is because we can easily produce global explanations based on the estimated coefficients of logistic regression models (LR) or by visualizing decision paths from decision tree models. Despite having high interpretability, these simple White-Box (WB) models have some weaknesses. For example, based on the intuition behind the LR classifier, the algorithm fails to build an accurate and reliable model if in reality the relationship between features and dependent variable (outcome) is non-linear or if there is interaction between features [57]. Fortunately, modifications on linear and additive models has created new algorithms such as Generalized Additive Models with pairwise interactions (GA^2M) [9] which is as accurate as BB models and still inherently interpretable. Explainable Boosting Machine (EBM) [9–11], a tree-based GA^2M model, will be used in this paper.

Black-Box (BB) models, on the other hand, do not generate natural interpretations. Instead, interpretations generally have to be extracted in a follow-up phase, after the model has been learned, using post-hoc explanation methods that extract information from learned black-box models such as ensemble methods [58] or neural networks, and help us to figure out "what else the model can tell us" [59]. These explanations are obtained by learning a separate interpretable model on the predictions of the black-box model [60, 61],

by perturbing inputs and analyzing the black-box model reactions [62], or by applying both methods [63]. The aim of these methods is to approximate the initial model (usually a BB model) to provide explanations [57]. Recent post-hoc approaches include Permutation Feature Importance [64], Local Interpretable Model-agnostic Explanations ($LIME$) [63], SHapley Additive exPlanations ($SHAP$) [65], Tree-SHAP (a variation of SHAP for tree-based ML models) and TreeInterpreter (TI) [66, 67].

2.3.1 Explainable Boosting Machine (EBM)

EBM [9–11] is an interpretability algorithm based on the Generalized Additive Model (GAM) [68, 69]. This glass-box model provides a high accuracy comparable to black-box models such as Random Forest and Boosted Trees while keeping its inherent interpretability [11]. Moreover, unlike black-box explanation approaches which are approximations of the model, EBM explanations are exact [57].

The standard GAMs have the form

$$g(E[y]) = \beta_0 + \sum f_j(x_j) \tag{1}$$

which adds single-feature models, called shape functions (f_j), through a linear function [69].

GA^2Ms model pairwise interactions between features (f_{ij}) to boost the accuracy compared to the standard GAMs [9].

$$g(E[y]) = \beta_0 + \sum f_j(x_j) + f_{ij}(x_i, x_j). \tag{2}$$

In the above formulas, x is an input, y is the label, and g is a link function.

EBM first learns the feature function f_j for the j^{th} feature, by using modern ML algorithms such as gradient boosting and bagging while considering only the j^{th} feature at a time and in a round-robin manner for all features. EBM also includes f_{ij} which is a pairwise interaction feature function to increase the accuracy of standard GAMs while maintaining the interpretability [11, 57]. By plotting the output value of one-dimensional (f_i) and two-dimensional (f_{ij}) components against their input values, we can visualize the

contribution of the features in shaping the final decision by the model. This is how GAMs provide interpretability [9].

2.3.2 Logistic Regression

Logistic Regression [56] is a statistical method that calculates the probabilities for binary classification problems. It aggregates the input features (x_i) with coefficients (β_i) in a linear way to predict the probability of an event (p).

$$\log \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + ... + \beta_n x_n. \tag{3}$$

The LR is a white-box model because the contribution of each input feature (x_i) in the model's output is interpretable. In other words, a unit increase in one of the input features (x_i) causes a log-odds output increase by one coefficient (β_i) [70]. Therefore, by interpreting the input response terms ($\beta_i x_i$), we can understand the output of the model.

2.3.3 SHapley Additive exPlanations (SHAP) and Tree-SHAP

SHAP is an additive feature attribution method and presents the explanations in a linear function which makes it more understandable for the users [54]. Shapley values explain the model's output of a function f as a sum of the effect ϕ_i that each feature has contributed to the output. Based on the additive feature attribution, the explanation model of g is defined as:

$$g(z\,') = \phi_0 + \sum_{i=1}^{M} \phi_i z\,'_i \tag{1}$$

Where M is the number of features, $z\,' \in \{0,1\}^M$, and $\phi_i \in \mathbf{R}$.

The $z\,'_i$ is equal to one if a feature being observed and equal to zero for unknown ones and the ϕ_i's are the attribution of features. $f(h_x(z\,'))$ is the mapping function that evaluate the effect of feature observation. Variable S is the set of non-zero indexes in z and $f_x(S) = f(h_x(z\,')) = E[f(x)|x_s]$. Here $E[f(x)|x_s]$ is the expected value of the function conditioned on a subset S of the input features. SHAP values combine these conditional expectations with the classic Shapley values from game theory to attribute ϕ_i values to each

Methods	Model Type			Final Explanation		Additive feature attribution method	Concept
	Model-specific	Model-agnostic	Surrogate	Local	Global		
Tree-SHAP (SHapley Additive exPlanations)	✓		✓	✓	✓	✓	Game theory Shapley values
LIME (Local Interpretable Model-agnostic Explanations)		✓	✓	✓		✓	Local approximation of the original predictive ML model
TreeInterpreter (TI)	✓			✓		✓	Decision Tree operations

Figure 2.1: Summary of Post-hoc Explanation Methods.

feature:

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(M - |S| - 1)!}{M!} [f_x(S \cup \{i\}) - f_x(S)] \tag{2}$$

where N is the set of all input features [71].

SHAP values can be extended to the SHAP interaction values based on the Shapley interaction index from game theory [72]. This approach discovers important pairwise interactions learned by the model. It shows if the effect of a feature on the predicted result depends on the value of other features.

$$\phi_{i,i} = \phi_i - \sum_{j \neq i} \phi_{i,j} \tag{2.9}$$

The SHAP interaction value between feature i and feature j is equal ($\phi_{ij} = \phi_{ji}$) and the total interaction effect is computed by adding $\phi_{ij} + \phi_{ji}$. Thus, the difference between the SHAP value (ϕ_i) and the SHAP interaction values for a feature (ϕ_{ij}) quantifies the main effects for a prediction ($\phi_{i,i}$).

In our study, we use Tree-SHAP which is a variant of SHAP for tree based machine learning models such as the LightGBM classifier [73] and Random Forest [51]. While Tree-SHAP adds local explainability to any model, it also adds a computational overhead.

A comparison of these three post-hoc explanations methods is presented in (Fig. 2.1).

2.3.4 Local Interpretable Model-agnostic Explanations (LIME)

LIME [63] is a model-agnostic approach which means it can be applied to any ML model and it provides individual explanations. To understand why a ML model made a specific prediction, LIME approximates the prediction of the ML model (f) with a linear regression explanation model (g) locally around that prediction (π_x) [57]. This explanation model can be achieved as follows:

$$\zeta(x) = \underset{g \in G}{\operatorname{argmin}} L(f, g, \pi_x) + \Omega(g) \tag{4}$$

The explanation model g belongs to a family of interpretable models (G) such as decision trees or linear models. The goal is to minimize $L(f, g, \pi_x)$ while keeping the complexity of the explanation model ($\Omega(g)$) low to achieve understandable explanations [63].

2.3.5 TreeInterpreter (TI)

Another post-hoc explanation method is TreeInterpreter that generates explanations for each individual predictions (local explanations) in tree-based models (either for regression or classification tasks) [66, 67]. This method follows the decision paths of each prediction's output, from the root to the leaf nodes in a tree, and extracts the available information for a specific feature, such as the expected prediction values at each internal and leaf node. Using this information, TI breaks down the final prediction value into a sum of feature contributions and a bias term (which is the mean over the training dataset). Therefore, TI makes the predicted output of the tree-based Black-Box models, such as RF, explainable and more understandable [66,67]. Thus the decision function $f(x)$ in a decision tree can be decomposed into the sum of feature contributions and a constant value of C_{full} (bias term) as formulated in Eq.2.10 [66, 67]:

$$f(x) = C_{full} + \sum_{k=1}^{K} contrib(x, k) \tag{2.10}$$

Here the value of the root node is defined as C_{full} (bias term) and x is an input feature vector. Thus $contrib(x, k)$ presents the contribution of the k-th feature in the feature vector x [66, 67].

Since a RF model builds several trees and takes the average of predictions made by those trees as its final predictions, the prediction function can be defined as an average of each function $f(x)$ from Eq.2.10. Therefore the decision function ($F(x)$) in a RF model can be calculated as follows:

$$F(x) = \frac{1}{J}\sum_{j=1}^{J} f_j(x) \tag{2.11}$$

$$F(x) = \frac{1}{J}\sum_{j=1}^{J} C_{j\,full} + \sum_{k=1}^{K}(\frac{1}{J}\sum_{j=1}^{J} contrib_j(x,k)) \tag{2.12}$$

Having a forest with J trees, the final prediction is decomposed by taking the average of both the bias term (C_{full}) and the contribution of each feature ($contrib(x,k)$) [66, 67].

2.3.6 Feature Importance in Decision Tree and Random Forest Classifiers

Since in our study we compare the global explanations between Decision Trees (DT) [55] and Random Forest (RF) [51], in this section we provide a brief review on how these explanations can be measured in different ML models.

The global explanation shows the importance of each feature (globally) on the model's outcome over the training set. The explanations are based on Feature Importance (FI) score that measures the impact of each feature on predicting the output of ML models (either Black Box or White Box models) such as RF or Logistic Regression (LR) [56]. However, the FI can not be considered as a "consistent" explainer [71]. It means that changing the model can decrease the importance of a feature despite the fact that the feature may still have a high impact on the predicted output [71].

In linear ML models such as LR, the output of the model is the weighted sum of the input features. Therefore, these linear models provide a set coefficients which can be used as the FI score. In tree-based models such as DT and RF, the importance of each feature is measured based on the reduction in impurity criterion after choosing that feature (i) to split a node (j) that is explained in Eq. 2.13 in more details.

In our experiments in Chapter 4, we will use Scikit-learn library [74] for building the models and comparing their global explanations based on the provided FI scores by

this library. The Scikit-learn implemented an optimized version of CART algorithm [75] (Classification and Regression Trees) for DT models. Therefore, the feature importance of DT models can be measured by Gini Impurity or Entropy [75] for classification tasks, and Mean Squared Error (MSE) or Mean Absolute Error (MAE) [76] for regressions. Considering a binary tree built by CART algorithm, the importance of node j after splitting on feature i (ni_j) is defined by calculating the reduction in node impurity (C_j) and weighting it (w_j) by the probability of reaching node j as explained in Eq.2.13 [77].

$$ni_j = w_j C_j - w_{\text{left}(j)} C_{\text{left}(j)} - w_{\text{right}(j)} C_{\text{right}(j)} \tag{2.13}$$

Here $C_{\text{left}(j)}$ and $w_{\text{left}(j)}$ are the impurity and the weighted number of samples reaching a child node from the left split on node j, respectively. The same notation is used for a child node from the right split on node j ($right(j)$) [77].

Then the importance of feature i (fi_i) is determined by adding the importance of all nodes j that were split on feature i, over the summation of node importance for all nodes k (ni_k) as: [77]

$$fi_i = \frac{\sum_{j:\text{node j splits on feature i}} ni_j}{\sum_{k \in \text{all nodes k}} ni_k} \tag{2.14}$$

Similarly, the RF algorithm implemented by Scikit-learn provides the feature importance score (fi_i) by averaging over all the trees as shown in Eq.2.15:

$$RFfi_i = \frac{\sum_{t:\text{all trees}} normfi_{it}}{T} \tag{2.15}$$

Where $normfi_{it}$ is the normalized importance of feature i in tree t and has values between 0 and 1. Finally, the feature importance in RF is defined by dividing the summation of normalized feature importance score on each tree t to the total number of trees T. Therefore, more important features have higher values [77].

2.3.7 Rank Similarity Metrics

To measure the similarity between the ranked lists of explanations (lists of the features that are ranked based on their contributions or impacts on the model's output) gener-

ated by Post-hoc methods, we use two types of metrics: Correlation based and Intersection (set) based.

2.3.7.1 Kendall's Tau correlation coefficient

One of the most commonly used rank correlation measures is Kendall's Tau correlation coefficient [78]. This metric calculates the probability of two items appearing in the same order in two (conjoint) ranking lists. If the items are in the similar order (identical), then correlation is strong and positive (+1). Whereas the reverse order results a strong and negative correlation (-1). Moreover, "randomly" related or uncorrelated items will have zero correlations coefficient [79, 80]. In calculating the correlations between every pair of items from two ranked lists, Kendall's Tau assumes that the lists are conjoint (all the features are available in both lists) and there are no ties between the lists (no two items have the same rank in each of two lists) [79]. Considering these assumptions, Kendall's Tau (τ) can be calculated as:

$$\tau = p_c - p_d = \frac{C}{P(n)} - \frac{D}{P(n)} = \frac{C - D}{\frac{n(n-1)}{2}} \tag{2.16}$$

Having n items in a ranked list, $P(n)$ is the total number of pairs. Therefore, the probability of choosing a pair of items (i, j) at random, if the pair have the same order in both ranked lists (concordant pairs), can be calculated as p_c. Similarly, the probability of discordant pairs (D) can be computed as p_d. Here C is the number of concordant pairs and D is the number of discordant pairs [79].

Despite the fact that Kendall's Tau has been widely used in quantifying the similarities of ranked lists, this metric is inefficient for non-conjoint rankings (when some items exist in one ranked list but not in the other list) [79]. Also, it does not assign higher weights to the top items of the ranked list (it is unweighted).

To address these issues, several approaches and implementations of the original Kendall's Tau (such as Kendall's Tau_a) have been proposed. In our experiments we will use one of the implementations from the Scipy statistical module[1] which by default calculates Kendall's Tau_b. This version of the Kendall's Tau handles the ties between ranked items

(when two items have the same rank). We applied Kendall's Tau_b with the default values for all the parameters.

Moreover, we will use a weighted Kendall's Tau implementation from the same statistic library[2]. In the weighted version of Kendall's Tau, the exchange of items with higher weights has more impact than the exchange of items with lower weights. Therefore, to assign more weights to the top-n items/features, we set the parameter *rank* as False to rank the items based on their indices directly. Another parameter is *weigher* which by default assigns a weight of $\frac{1}{1+r}$ to each item with rank r.

2.3.7.2 Rank-Biased Overlap (RBO)

Among the intersection based similarity measures, we chose RBO (Rank-Biased Overlap) [79] that supports the Kendall's Tau issues. The RBO measures the similarity between two rankings even if they are incomplete (which in that case needs to support non-conjoint rankings) and at any depth of the lists. Looking at RBO from the set intersection (overlap) perspective, the overall idea behind RBO is to "bias the proportional overlap at each depth by a convergent series of weights" (a geometric series with a finite sum as formulated in Eq. 2.17) [79]. In other words, RBO can be calculated by averaging the proportion of overlapping items while increasing the depths.

$$\sum_{d=1}^{\infty} p^{(d-1)} = \frac{1}{1-p} \tag{2.17}$$

Where d is the depth of the ranking list that are being tested and p defines the degree of contributions that the top-d items could have while measuring the similarity by *RBO*.

Having two infinite ranking lists of S and T, the S_i and T_i refer to their elements at rank i. Therefore, the size of the intersection (overlap), $(I_{S,T,d})$, between these two ranking lists at each depth d can be measured as:

$$I_{S,T,d} = S_{:d} \cap T_{:d} \tag{2.18}$$

[1]https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.kendalltau.htmlscipy.stats.kendalltau
[2]https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.weightedtau.htmlscipy.stats.weightedtau

$$X_{S,T,d} = |I_{S,T,d}| \tag{2.19}$$

Where $S_{:d}$ and $T_{:d}$ refer to the elements that appear in the intersection of list S and T, from position 1 to d. Defining A_d as the proportion of the items overlapping at depth d, the average of the overlap ($AO(S,T,k)$) can be formulated as shown in Eq. 2.18 and 2.19:

$$A_d = \frac{X_{S,T,d}}{d}, \tag{2.20}$$

$$AO(S,T,k) = \frac{1}{k}\sum_{d=1}^{k} A_d. \tag{2.21}$$

Here k is the evaluation depth. After having the average of overlaps at each depth, considering w as a vector of weights and w_d as the weight at position d, the similarity between two ranked lists S and T can be measured using Eq. 2.22 [79]:

$$SIM(S,T,w) = \sum_{d=1}^{\infty} w_d \cdot A_d \tag{2.22}$$

If w is a convergent vector of weights (with finite sum as formulated in Eq. 2.17), then the Rank-biased overlap can be derived as [79]:

$$RBO(S,T,p) = (1-p)\sum_{d=1}^{\infty} p^{(d-1)} \cdot A_d \tag{2.23}$$

Therefore, RBO ranges from 0 to 1, where 0 means the ranked lists are disjoint, and 1 means they are identical. In Eq. 2.23, the degree of contribution that the top-d items can have on the RBO is defined by the parameter p. Thus, the smaller value of p indicates that the top ranked items have more weights in measuring similarities by RBO compared to the items in the bottom of the lists. Similarly, for the values of p closer to 1, the weights become arbitrarily flat [79] and RBO becomes unweighted.

2.4 Summary

In this chapter, we started with reviewing the literature about recommendation systems and existing concerns about transparency and bias in the recommendations. We continued our discussion about the role of trust, transparency and explainability in making

TABLE 2.1: Rank similarity metrics' range of values

	-1	0	1
Kendall's Tau	similar in reverse order	disjoint (no similarity)	identical (similar)
Weighted Kendall's Tau	similar in reverse order	disjoint	identical
RBO	NA	disjoint	identical
Weighted RBO	NA	disjoint	identical

more practical and effective recommender systems. Then, we reviewed sources of bias in the recommendation process. We ended the recommender systems section by describing evaluation metrics that we plan to use to evaluate the performance of our proposed methods in terms of explainability, novelty and diversity in Chapter 3.

Moreover, we reviewed previous work in making reliable robotic grasps. Despite advances in robot control, imprecision in sensing and actuation still is a challenge in making a robot's grasp more stable. Furthermore Black Box models can fail to explain the reasons for predicted failures, and thus give no clues about why failures occur, whether to trust the prediction of failure, or how to avoid failures for instance by different designs. This is why explainability in ML could provide a solution for the failure prediction challenge.

We also reviewed several interpretable ML methods, which can generate global and local explanations, including post-hoc explanation methods.

Finally, we discussed criteria (such as feature importance) and metrics (such as Kendall's Tau and RBO) that we plan to use to compare the consistency of the similarity between the generated (global or local) explanations from different explanation methods in Chapter 4.

CHAPTER 3

BEYOND ACCURACY IN MACHINE LEARNING: TRANSPARENCY AND BIAS TRADE-OFFS IN COLLABORATIVE FILTERING RECOMMENDATION SYSTEMS

In this chapter, we study ML criteria that go beyond accuracy in collaborative filtering recommendation, where we study explainability and bias in addition to accuracy. In Section 3.1, we propose different algorithms to mitigate the effect of popularity debiasing on explainability and accuracy of *Explainable Matrix Factorization (EMF)* model. Then in Section 3.2, we present the results for the proposed Debiased EMF based methods, while comparing them with baseline methods, and evaluating their performance from different perspectives; namely *accuracy*, *explainability*, *novelty*, and *diversity*.

3.1 Debiased Explainable Matrix Factorization (Debiased EMF)

As we mentioned in Chapter 2, inverse propensity scoring (IPS) is one of the most popular methods to mitigate the effect of popularity of items on model training. The IPS under-weights popular items to increase the chance of rare (non-popular) items to be recommended to users.

In propensity-based MF [27], the IPS is used in the objective function to under-weight the loss for each observation as shown in Eq. 3.1:

$$\underset{V,M}{\operatorname{argmax}} \sum_{O_{u,i}=1} \frac{\parallel R - V^T M \parallel^2}{P_{u,i}} + \lambda(\parallel V \parallel_F^2 + \parallel M \parallel_F^2) \tag{3.1}$$

Here $P_{u,i}$ is the propensity score which shows the probability that a user u will see item i. V and M are the two latent factors in the MF. $O_{u,i}$ is a binary value which indicates that if user u provided a rating for item i to the system. ($O_{u,i}$ equal to 1 means that user

u has observed item i and provided rating R.)

One way to estimate the *propensity* is to use a *popularity score* [81]. This method assumes that $O_{u,i}$ has a *Bernoulli* distribution and the propensity score is fixed among users. Therefore, having a rating matrix, the popularity of an item is the proportion of that item being exposed to certain users among all the users [82]. In our experiment, we used the popularity score to calculate the IPS as shown by Eq. 3.2:

$$P_{u,i} = \frac{N_i}{N_U} \tag{3.2}$$

Here $P_{u,i}$ is the probability that user u has observed item i and is equal to the ratio of N_i (number of users who rated item i) to N_U (total number of users).

To study the effect of popularity bias on explainability and accuracy of a MF-based model, we applied the IPS on the state-of-the-art *Explainable Matrix Factorization* (EMF) [4, 5]. EMF is a latent factor model which adds a soft explainability constraint to the *Matrix Factorization* (MF) objective function (Eq. 3.3) while increasing the accuracy of the model (resulting in Eq. 3.4).

$$J_{MF} = \sum_{u,i \in R} (r_{u,i} - p_u q_i^T)^2 + \frac{\beta}{2}(\parallel p_u \parallel^2 + \parallel q_i \parallel^2) \tag{3.3}$$

$$J_{EMF} = \sum_{u,i \in R} (r_{u,i} - p_u q_i^T)^2 + \frac{\beta}{2}(\parallel p_u \parallel^2 + \parallel q_i \parallel^2) + \frac{\lambda}{2} \parallel p_u - q_i \parallel^2 W_{u,i} \tag{3.4}$$

Here $r_{u,i}$ is the rating that user u has given to item i. p_u and q_i are representations for user u and item i in a lower dimension (k) and their dot product ($p_u q_i^T$) results in an approximation for the rating that user u has given to item i. β is a coefficient for L_2 regularization term ($\frac{1}{2}(\parallel p_u \parallel^2 + \parallel q_i \parallel^2)$). The parameter λ is a coefficient to control the softness of learning new representations (p_u and q_i) in explainability regularization term ($\frac{\lambda}{2} \parallel p_u - q_i \parallel^2 W_{u,i}$). λ also manages the trade-offs between accuracy and explainability [4]. $W_{u,i}$ is an *explanation matrix* which can be calculated in different formats. In our study we use *Neighborhood Style Explanation* (NSE), which provides explanations based on similar users, used by [4]. The goal in EMF is to recommend explainable items i to user u by

learning their representations in the latent space (p_u and q_i) that are close to each other. Thus, p_u- q_i should be close to zero.

Eq. 3.5 defines the explainability matrix $W_{u,i}$ which calculates the explainability score of explainable items ($Expl_{u,i} \geq \sigma$) between each pair of users and items. We chose the explainability threshold σ as zero based on the previously recommended in [4, 5].

$$W_{u,i} = \begin{cases} Expl_{u,i} & if\ Expl_{u,i} \geq \sigma \\ 0 & otherwise \end{cases} \tag{3.5}$$

The NSE_based explainability is calculated as the expected value of the ratings given on item i by users who are similar to user u [4]:

$$\mathbf{E}(r_{u,i}|N_u) = \sum_{z \in \zeta} z \times \mathbf{P}(r_{u,i} = z|N_u) \tag{3.6}$$

Given a set of similar users for user u (N_u), the probability of giving rating z to item i can be estimated by following equation:

$$\mathbf{Pr}(r_{u,i} = z|N_u) = \frac{|N_u \cap U_{i,z}|}{|N_u|} \tag{3.7}$$

Here $U_{i,z}$ is the set of users who have given rating z to item i. Thus, the probability is equal to the proportion of number of users who have given rating z to item i, and are similar to user u ($|N_u \cap U_{i,z}|$) to the number similar users for user u ($|N_u|$) .

Analysing the sensitivity of EMF to explainability and accuracy, [4, 5] found that there is a positive correlation between explainability and accuracy. They showed that by increasing the explainability parameter (λ), there is no sacrifice in the accuracy of EMF which in fact increases. However the explainability scores which aggregate the neighbors' ratings may be affected by popularity bias.

To mitigate the effect of popular items on the model's explainability and accuracy, we propose a *Debiased Explainable Matrix Factorization* (Debiased EMF) method which uses IPS in its model training. The *inverse popularity propensity score* ($P_{u,i}$) can be applied to in two different terms of the EMF objective in Eq. 3.4:

1. rating prediction loss ($(r_{u,i} - p_u q_i^T)^2$)

2. explainability regularization ($\frac{\lambda}{2} \parallel p_u - q_i \parallel^2 W_{u,i}$)

Therefore, we investigate three *Debiased EMF* models.

The first model is *Debiased_EMF_1*, which uses inverse popularity propensity score in its rating prediction loss as shown in Eq. 3.8:

$$J_{Debiased\_EMF\_1} = \sum_{u,i \in R} \frac{(r_{u,i} - p_u q_i^T)^2}{P_{u,i}} + \frac{\beta}{2}(\parallel p_u \parallel^2 + \parallel q_i \parallel^2) + \frac{\lambda}{2} \parallel p_u - q_i \parallel^2 W_{u,i} \quad (3.8)$$

Algorithm 3.1 Debiased Explainable Matrix Factorization 1 (Debiased_EMF_1)

Input: data matrix R, number of factors k, number of neighbors n, hyperparameter α, β, and λ, *top_n* recommended item list. **Output:** P and Q

1. **for** each user u**:**
   (a) calculate $N_n(u)$ using Cosine similarity
2. **end for**
3. **for** each user-item pair (u, i)**:**
   (a) calculate $W_{u,i}$ using Eq. 3.5
4. **end for**
5. **for** each item i:
   (a) calculate the ratio of $\frac{N_i}{N_U}$
6. **end for**
7. **for** each user_item pair (u, i):
   (a) calculate $P_{u,i}$ using Eq. 3.2
8. **end for**
9. initialize P_u and Q_i
10. **for** each $r_{u,i}$ from the training data**:**
    (a) solve for p_u^{new} and q_i^{new} using the update rule in Eq. 3.11
11. **end for**

The idea behind *Debiased_EMF_1* is to down-weight popular items in rating prediction and to provide more explainable items.

In the second model, which is called $Debiased\_EMF\_2$, the explainability regularization term is multiplied by the inverse popularity propensity score as shown in Eq. 3.9:

$$J_{Debiased\_EMF\_2} = \sum_{u,i \in R} (r_{u,i} - p_u q_i^T)^2 + \frac{\beta}{2}(\parallel p_u \parallel^2 + \parallel q_i \parallel^2) + \frac{\lambda}{2} \parallel p_u - q_i \parallel^2 \frac{W_{u,i}}{P_{u,i}} \quad (3.9)$$

In $Debiased\_EMF\_2$, the intuition is the more explainable item i is to user u and the less popular item i is for user u, the closer is user's and item's representation in latent space to each other. In other word, the goal is to make user's representation in latent space (p_u) to be close to more explainable and less popular item (q_i), and thus enhance the chance for those items to be recommended.

Finally, in $Debiased\_EMF\_1\_2$, we apply the IPS not only in the rating prediction loss, but also in the explainability regularization term. Thus, we increase the weight of less popular items in both the prediction and explanation components. (Eq. 3.10)

$$J_{Debiased\_EMF\_1\_2} = \sum_{u,i \in R} \frac{(r_{u,i} - p_u q_i^T)^2}{P_{u,i}} + \frac{\beta}{2}(\parallel p_u \parallel^2 + \parallel q_i \parallel^2) + \frac{\lambda}{2} \parallel p_u - q_i \parallel^2 \frac{W_{u,i}}{P_{u,i}} \quad (3.10)$$

We use stochastic gradient descent to minimize the objective functions for the proposed methods. The updated p_u and q_i can be obtained by differentiation and are given by the following Equations for $Debiased\_EMF\_1$, $Debiased\_EMF\_2$, $Debiased\_EMF\_1\_2$ respectively :

Updated equation for $Debiased\_EMF\_1$:

$$\begin{aligned} \boldsymbol{p_u}^{(t+1)} &\leftarrow \boldsymbol{p_u}^{(t)} + \alpha(2\frac{(r_{u,i} - p_u q_i^T)}{P_{u,i}}\boldsymbol{q_i} - \beta \boldsymbol{p_u} - \lambda(\boldsymbol{p_u} - \boldsymbol{q_i})W_{u,i}) \\ \boldsymbol{q_i}^{(t+1)} &\leftarrow \boldsymbol{q_i}^{(t)} + \alpha(2\frac{(r_{u,i} - p_u q_i^T)}{P_{u,i}}\boldsymbol{p_u} - \beta \boldsymbol{q_i} + \lambda(\boldsymbol{p_u} - \boldsymbol{q_i})W_{u,i}) \end{aligned} \quad (3.11)$$

Updated equation for $Debiased\_EMF\_2$:

$$\begin{aligned} \boldsymbol{p_u}^{(t+1)} &\leftarrow \boldsymbol{p_u}^{(t)} + \alpha(2(r_{u,i} - p_u q_i^T)\boldsymbol{q_i} - \beta \boldsymbol{p_u} - \lambda(\boldsymbol{p_u} - \boldsymbol{q_i})\frac{W_{u,i}}{P_{u,i}}) \\ \boldsymbol{q_i}^{(t+1)} &\leftarrow \boldsymbol{q_i}^{(t)} + \alpha(2(r_{u,i} - p_u q_i^T)\boldsymbol{p_u} - \beta \boldsymbol{q_i} + \lambda(\boldsymbol{p_u} - \boldsymbol{q_i})\frac{W_{u,i}}{P_{u,i}}) \end{aligned} \quad (3.12)$$

Updated equation for $Debiased\_EMF\_1\_2$:

Algorithm 3.2 Debiased Explainable Matrix Factorization 2 (Debiased_EMF_2)

Input: data matrix R, number of factors k, number of neighbors n, hyperparameter α, β, and λ, *top_n* recommended item list. **Output:** P and Q

1. **for** each user u**:**
   (a) calculate $N_n(u)$ using Cosine similarity
2. **end for**
3. **for** each user-item pair (u, i)**:**
   (a) calculate $W_{u,i}$ using Eq. 3.5
4. **end for**
5. **for** each item i:
   (a) calculate the ratio of $\frac{N_i}{N_U}$
6. **end for**
7. **for** each user_item pair (u, i):
   (a) calculate $P_{u,i}$ using Eq. 3.2
8. **end for**
9. initialize P_u and Q_i
10. **for** each $r_{u,i}$ from the training data**:**
    (a) solve for p_u^{new} and q_i^{new} using the update rule in Eq. 3.12
11. **end for**

$$\begin{aligned} \boldsymbol{p_u}^{(t+1)} &\leftarrow \boldsymbol{p_u}^{(t)} + \alpha(2\frac{(r_{u,i}-p_u q_i^T)}{P_{u,i}}\boldsymbol{q_i} - \beta\boldsymbol{p_u} - \lambda(\boldsymbol{p_u}-\boldsymbol{q_i})\frac{W_{u,i}}{P_{u,i}}) \\ \boldsymbol{q_i}^{(t+1)} &\leftarrow \boldsymbol{q_i}^{(t)} + \alpha(2\frac{(r_{u,i}-p_u q_i^T)}{P_{u,i}}\boldsymbol{p_u} - \beta\boldsymbol{q_i} + \lambda(\boldsymbol{p_u}-\boldsymbol{q_i})\frac{W_{u,i}}{P_{u,i}}) \end{aligned} \tag{3.13}$$

The algorithms for the proposed methods are described in Algorithms 3.1, 3.2, and 3.3. In the next section, we will present experimental results to compare the effect of IPS-based debiasing in the proposed models.

Algorithm 3.3 Debiased Explainable Matrix Factorization 1 and 2 (Debiased_EMF_1_2)

Input: data matrix R, number of factors k, number of neighbors n, hyperparameter α, β, and λ, top_n recommended item list. **Output:** P and Q

1. **for** each user u**:**
   (a) calculate $N_n(u)$ using Cosine similarity
2. **end for**
3. **for** each user-item pair (u, i)**:**
   (a) calculate $W_{u,i}$ using Eq. 3.5
4. **end for**
5. **for** each item i:
   (a) calculate the ratio of $\frac{N_i}{N_U}$
6. **end for**
7. **for** each user_item pair (u, i):
   (a) calculate $P_{u,i}$ using Eq. 3.2
8. **end for**
9. initialize P_u and Q_i
10. **for** each $r_{u,i}$ from the training data**:**
    (a) solve for p_u^{new} and q_i^{new} using the update rule in Eq. 3.13
11. **end for**

3.2 Experimental Evaluation of Accuracy, Explainability and Bias Trade-offs in Matrix Factorization based Recommender Systems

We used the public MovieLens benchmark ratings data [83]. The dataset includes 100,000,000 ratings, on a scale of 1 to 5, for 1682 movies and 943 users. We split the data randomly into 80% for training, 10% for validation and model hyperparameter tuning, and 10% for testing and reporting the generalization of the model.

We tuned the hyperparameters of each model using grid search, while optimizing the model accuracy on the validation set by finding the best MAP. We performed all the experiments 10 times and report the average of the results. The values for the learning

TABLE 3.1: Accuracy Evaluation of the Models

Model	*k*	*alpha*	*beta*	*lambda*	*RMSE*	*MAE*	*NDCG*	*MAP*
MF	25	0.001	0.0001	0	1.08492	0.83843	0.89455	**0.00738**
EMF	20	0.0001	0.001	0.001	<u>1.05210</u>	<u>0.81064</u>	<u>0.90061</u>	<u>0.00723</u>
Popularity_Propensity_MF	50	0.0001	0.01	0	2.96409	1.90082	0.85813	0.00340
Debiased_EMF_1	50	0.0001	0.0001	0.001	9.17890	8.97475	0.83186	0.00184
Debiased_EMF_2	20	0.001	0.0001	0.001	**1.03119**	**0.79663**	**0.90284**	0.00693
Debiased_EMF_1_2	20	0.00001	0.0001	0.01	2.09869	1.69614	0.83390	0.00163

parameter (α), L_2 regularization (β), and explainability regularization (λ) were chosen from the following sets: α=[0.001, 0.0001, 0.00001], β=[0.01, 0.001, 0.0001], λ = [0.1, 0.01, 0.001]. We set the size of the recommendation list ($|top\_n|$) to 10 and the number of neighbors to 20 based on the previously recommended ranges in [4, 5]. Also, for counting the number of non-popular items (when evaluating diversity), we chose the popularity threshold (θ) to be 0.1. In this way if the popularity of an item is less than 0.1, we treat it as a non-popular item among the *top*_10 recommended item list.

Next, we ran the experiment to test the trained models on the test set by considering the best hyperparameters for each model and computed different evaluation metrics. Since we have tuned the hyperparameters based on MAP, we reported the results for each model with the highest MAP. For example, for MF the best MAP was obtained at k= 25, λ = 0.001, and β = 0.0001. Thus, we only show the evaluation metrics that were obtained with these parameters for MF.

Tables 3.1−3.3 compare the performance of the models based on their accuracy, explainability, novelty and diversity respectively. Each row shows different metrics for each model which were obtained by the aforementioned parameters in that row. The best metric's value for each model is in bold and the second to the best is underlined.

Evaluating the models based on their accuracy from Table 3.1, we see that *Debiased_EMF_2* surpassed other models in most rating prediction (accuracy and ranking quality) metrics followed by EMF [4, 5]. The only exception was in MAP, on which *MF* [3] showed better performance. A possible interpretation for this result could be due to the

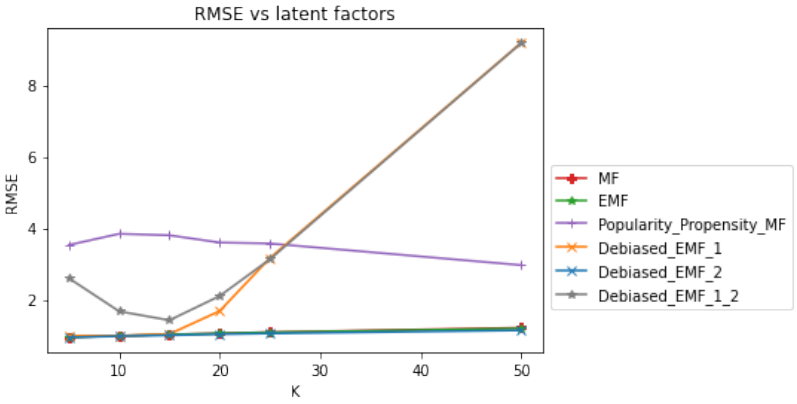

Figure 3.1: *Debiased_EMF_2* has the best accuracy performance in terms of *RMSE*.

optimization part in which we tuned the hyperparameters to get the best *MAP*, as is customary in the literature on training *MF* for recommendation application.

In terms of *RMSE* (*Root Mean Squared Error*), *MAE* (*Mean Absolute Error*), and *NDCG* (*Normalized Discounted Cumulative Gain*) , we can see that adding explainability (from *MF* to *EMF*) and then debiasing the explainability regularization part (in *Debiased_EMF_2*) has improved the accuracy of the rating predictions. A comparison in performance of other models such as *Popularity_Propensity_MF*, *Debiased_EMF_1*, and *Debiased_EMF_1_2*, shows a decrease in their rating prediction accuracy. According to these observations, debiasing the prediction loss by multiplying it by the *IPS* (inverse popularity propensity score) hurts the accuracy of the models, while using the *IPS* in the explainability regularization part improves the accuracy. This result illustrates the trade-offs between accuracy and popularity debiasing in the performance of the models.

Figures 3.1 and 3.2 also show the effect of varying the latent factor number (k) on the *RMSE* and *NDCG* values in each model, respectively. Fig. 3.2 shows that *MF*, *EMF*, and *Debiased_EMF_2* had lower errors (in terms of *NDCG*) in predicting unseen ratings while *Popularity_Propensity_MF*, *Debiased_EMF_1*, and *Debiased_EMF_1_2* had higher errors. *Debiased_EMF_1* had high accuracy up to the $k = 15$ latent factors. Then its accuracy decreased with increasing latent factors beyond 15. Fig. 3.2 presents changes in the *NDCG* values by increasing the latent factor (k) for each model. It shows that *Debiased_EMF_2*

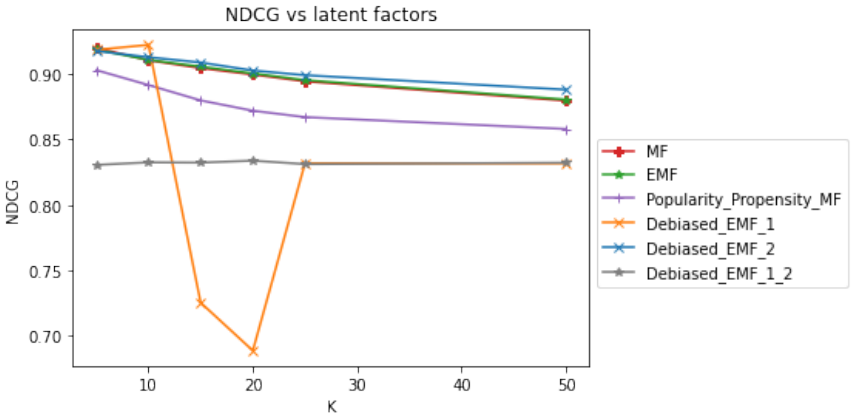

Figure 3.2: *Debiased_EMF_2* has the best ranking quality in terms of *NDCG*.

TABLE 3.2: Explainability Evaluation of the Models

Model	*k*	*alpha*	*beta*	*lambda*	*WMEP*	*TMEP*	*WTMEP*
MF	25	0.001	0.0001	0	1.60379	0.82050	1.44055
EMF	20	0.0001	0.001	0.001	1.61120	0.82314	1.44994
Popularity_Propensity_MF	50	0.0001	0.01	0	1.57358	0.80479	1.38932
Debiased_EMF_1	50	0.0001	0.0001	0.001	1.48487	0.77561	1.29055
Debiased_EMF_2	20	0.001	0.0001	0.001	**1.61656**	**0.82538**	**1.45806**
Debiased_EMF_1_2	20	0.00001	0.0001	0.01	1.49174	0.77757	1.29896

was the best among other models in terms of ranking quality by *NDCG*. For latent factors between 5 and 10, *Debiased_EMF_2* had the highest *NDCG*.

Table 3.2 compares the models based on their explainability. Here, also the *Debiased_EMF_2* model showed the best performance in terms of explainability metrics such as *WMEP*, *TMEP*, and *WTMEP* following by *EMF* [4,5]. Models with debiasing in their prediction loss (*Popularity_Propensity_MF*, *Debiased_EMF_1*, and *Debiased_EMF_1_2*) showed weaker performance in terms of explainability. These results also show the different impacts of popularity debiasing in the prediction loss and explainability regularization term and suggest a trade-offs between explainability and popularity debiasing in performance of the models. Fig. 3.3 shows that increasing the latent factor k decreases the *WMEP* values for most of the models. It also shows that *Debiased_EMF_2* had better performance in terms of explainability compared to the rest of models.

Table 3.3, which presents the evaluation of the models in terms of their novelty and

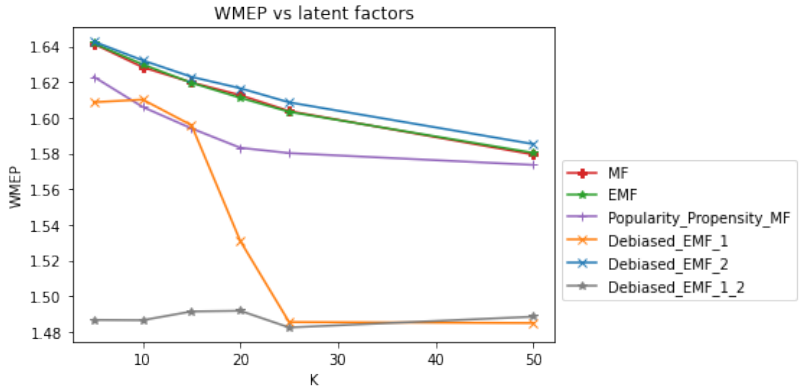

Figure 3.3: *Debiased_EMF_2* has the best explainability performance in terms of *WMEP*

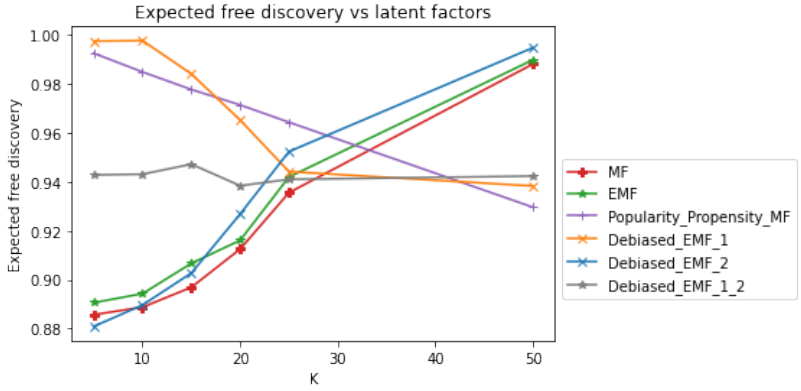

Figure 3.4: Users are expected to be able to see more novel items (compared to baselines) for both *Debiased_EMF_1_2* and *Debiased_EMF_1* models.

diversity metrics (*EFD*, *Diversity*), shows that *Popularity_Propensity_MF* was able to recommend more diverse (less popular) items to the users following by *Debiased_EMF_1*. In other word, on average 9.4, items from the list of *top_10* recommended items to users by *Popularity_Propensity_MF* were non-popular (diverse). Fig. 3.5 shows that by increasing the latent factor number k, models such as *MF*, *EMF*, and *Debiased_EMF_2* were able to recommend more non-popular items to users while for the rest of the models such as *Popularity Propensity MF* and *Debiased_EMF_1*, the diversity of the list of recommended item has been decreased.

In terms of *novelty*, the results show that users were recommended more novel items (compared to what they have previously observed) by *Debiased_EMF_1_2*. *Debiased_EMF_1* also showed a comparable results, in terms of novelty, to *Debiased_EMF_1_2*. Fig. 3.4

TABLE 3.3: Novelty and Diversity Evaluation of the Models

Model	*k*	*alpha*	*beta*	*lambda*	*EFD(Novelty)*	*Diversity*
MF	25	0.001	0.0001	0	0.93552	7.55642
EMF	20	0.0001	0.001	0.001	0.91608	6.83489
Popularity_Propensity_MF	50	0.0001	0.01	0	0.92960	**9.38706**
Debiased_EMF_1	50	0.0001	0.0001	0.001	<u>0.93826</u>	<u>7.94252</u>
Debiased_EMF_2	20	0.001	0.0001	0.001	0.92681	7.26363
Debiased_EMF_1_2	20	0.00001	0.0001	0.01	**0.93838**	7.65769

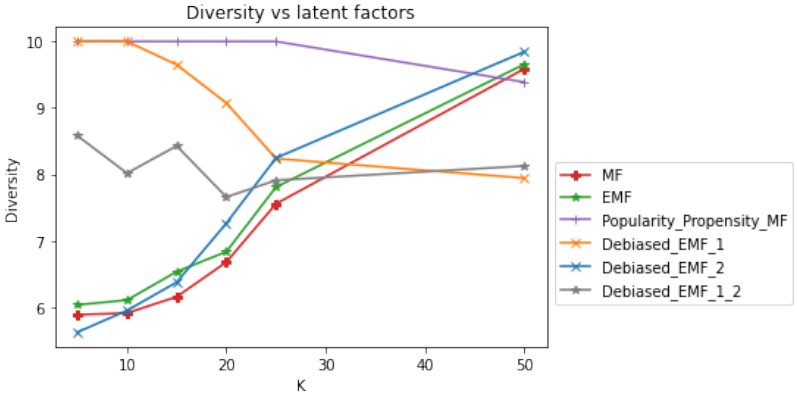

Figure 3.5: *Popularity_Propensity_MF* was able to recommend more diverse (less popular) items to the users following by *Debiased_EMF_1*.

shows that by increasing the latent factor number k, the novelty metrics (EFD) has been increased for MF, EMF, and *Debiased_EMF_2*, while it has been decreased for Popularity_Propensity_MF and *Debiased_EMF_1*. For *Debiased_EMF_1_2*, there were some changes for latent factor numbers ranging from 5 to 25, but after $k = 25$ its EFD values remained almost the same.

3.3 Summary

In this chapter, we presented several approaches to mitigate the effect of popularity debiasing on *Explainable Matrix Factorization (*EMF*)* model [4] by applying *inverse popularity propensity score* (*IPS*). We discussed the intuition behind our proposed methods in using the *IPS* in different terms of the *EMF* objective function. We also presented the experimental results for studying the effect of popularity debiasing on our proposed *EMF* based models in terms of accuracy, explainability, novelty and diversity.

The experiments showed that *Debiased_EMF_2*, which was debiased in its explainability regularization term, outperformed other models in most accuracy and explainability metrics. In terms of accuracy metrics such as *RMSE*, *MAE*, and *NDCG*, we can see that adding explainability (from *MF* to *EMF*) and then debiasing the explainability regularization part (from *EMF* to *Debiased_EMF_2*) has improved the accuracy of the rating predictions. Comparing the explainability of the models based on metrics such as *WMEP*, *TMEP*, and *WTMEP*, we found that *Debiased_EMF_2* had the best performance. While in terms of novelty and diversity, models with a debiasing term in their rating prediction loss such as *Popularity_Propensity_MF* [82], *Debiased_EMF_1*, and *Debiased_EMF_1_2* have better performance.

CHAPTER 4

BEYOND ACCURACY IN MACHINE LEARNING: TRANSPARENCY AND ACCURACY TRADE-OFFS IN ROBOTIC FAILURE PREDICTION

In this chapter, we study Machine Learning criteria that go beyond accuracy, in robotic grasp failure prediction, where we study *explainability* in addition to prediction *accuracy*. In Section 4.1, we present two methodologies to compare the *explainability* and *accuracy* of white-box and black-box ML models. Then we present another methodology to evaluate the explanations generated by different post-hoc explanation methods (Section 4.2). Finally in Section 4.1.1 and Section 4.2.1, we present the results of studying the trade-offs between *accuracy* and *explainability* in the robotic grasping failure prediction problem using three case studies, presented in Sections 4.1.2, 4.1.3, and 4.2.1.1, respectively.

4.1 A Methodology for Studying Accuracy and Explainability Trade-offs in Robotic Failure Prediction

Grasping reliability is important in many robotic tasks involving human safety or material costs. For this reason, predicting grasp failures before they occur can give timely warnings about potential reliability risks during manipulation. These predictions can guide decision making by both humans interacting with the robot and by design engineers seeking to improve the robot performance.

Traditional failure prediction, by using machine learning (ML) methods, utilizes both experiments and simulation task performance data. ML models include either White-Box (WB) models, whose prediction mechanism is interpretable, or Black-Box (BB) models who have superior performance, but are opaque to interpretation. In fact, black-box models do not have the inherent ability to explain the reasons behind their predictions.

In many cases, particularly those where robots must collaborate with humans, it is not sufficient to predict an impending failure. In addition, the predictions must come with explanations.

We present an empirical study of the trade-offs between prediction accuracy and explainability using the interpretability of white-box models and post-hoc explanations methods, such as Tree-SHAP [65, 71], to generate explanations from black-box models. For this purpose, we use the Kaggle grasp dataset which was generated with the Shadow Hand [1, 2].

We also investigate an alternative approach to accurate and yet explainable failure prediction. For this purpose, we start by contrasting the performance and explainability of grasp failure prediction using a classical interpretable white-box ML model and a more sophisticated glass-box model that shows a gain in performance without sacrificing interpretability. The glass-box model, Explainable Boosting Machine (EBM) [9–11], provides a high accuracy while being designed to be explainable. EBM is a type of Generalized Additive Model [68, 69] which models the individual features' impacts on the model's output, as well as the pairwise interaction between the features.

Furthermore, we investigate extracting local explanations, for individual test cases, using Local Interpretable Model-agnostic Explanations (LIME) [63] (the most popular post-hoc explanation method) and EBM [9–11] as an inherently explainable glass box model. *Local* explanations can provide contextual and specialized explanations for individual grasping cases, and thus help explain both white-box models and glass-box models, in addition to black-box models. Ideally, these explanations should be able to show how each feature has contributed (positively or negatively) to the predicted results for any *individual* sample in the data. This is in contrast to *global* explanations which cannot explain an individual instance's prediction, but rather give an overall impact of the features for an entire set of data or instances.

In the first study of grasp failure prediction and explainability, in Section 4.1.2, we explore the following research questions:

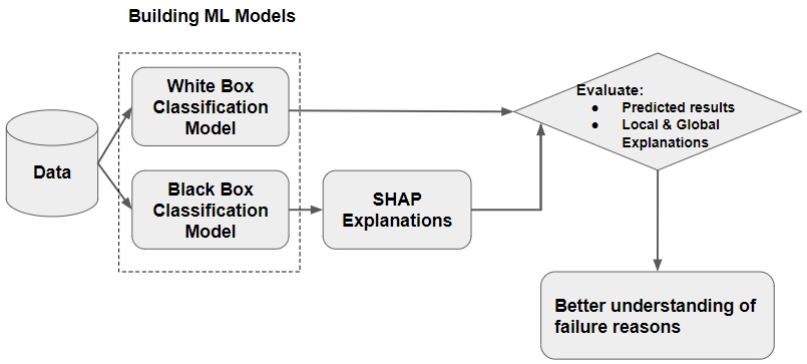

Figure 4.1: Methodology Flow Diagram 1 (white box and black box models).

- RQ1: How do white box and black box models perform in terms of accuracy and explainability on this particular task?
- RQ2: Can we achieve both explainability and accuracy in the grasp failure prediction task?

To answer these questions, we train different models on grasp failure simulation data, then compare the white-box models to black-box models. To bridge the black box model's accuracy and the white box model's explainability we extract local and global explanations for the black box model using the SHAP explanation method [65, 71] (Fig. 4.1).

We use Tree-SHAP [65,71] to explore local and global explanations and to determine which features are most responsible for grasp failures, in general (global explanations), or for a specific input test case (local explanation). For instance, which features (torque or velocity) and their values, in which joint and finger has contributed more in failing a grasp.

In the second grasp failure prediction and explainability study, in Section 4.1.3, we train and compare two different models on grasp failure simulation data, a white-box model (Logistic Regression [56]) which is inherently interpretable and a glass-box model (EBM [9–11]) which is designed to have a comparable accuracy to black-box models while maintaining its interpretability. Since the LR classifier does not explain individual predictions, we use the LIME [63] method to provide local explanations for any sample of data.

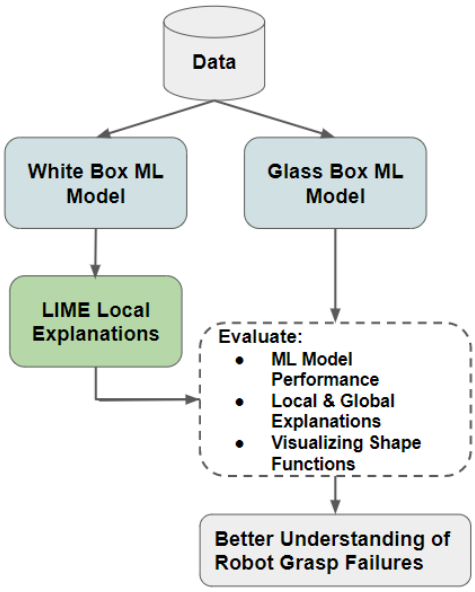

Figure 4.2: Methodology Flow Diagram 2 (White box and glass box models).

In addition to the local and global explanations, we visualize the shape function of one of the most important features in predicting the risk of robot grasp failures to monitor the feature's behavior in both models and to compare the ability of EBM and LR models in capturing non-linear and more complex relationships in the data. Fig. 4.2 depicts the flow of our methodology, starting with learning predictive models for failure, then generating explanations for predicted failures, and finally evaluating the results. The experimental results will be presented in Section 4.1.3.

4.1.1 Experimental Results for Studying Accuracy and Explainability Trade-offs in Robotics Failure Prediction

4.1.1.1 Dataset information

In the robotic grasp failure prediction task, we used the simulated robot grasp dataset provided by the author [1] and publicly is available in the Kaggle [2]. This dataset was generated by using the Smart Grasping Sandbox simulation for the Shadow's Smart Grasping System[1] with the UR10 robot from the Universal Robots company[2]. The end-effector had three fingers and each finger had three joints. In total, there were nine joints and in each

TABLE 4.1

Notation of the input features

Feature	Definition
$H_1\_F_1J_1\_eff$	effort in joint 1 in finger 1
$H_1\_F_1J_1\_vel$	velocity in joint 1 in finger 1

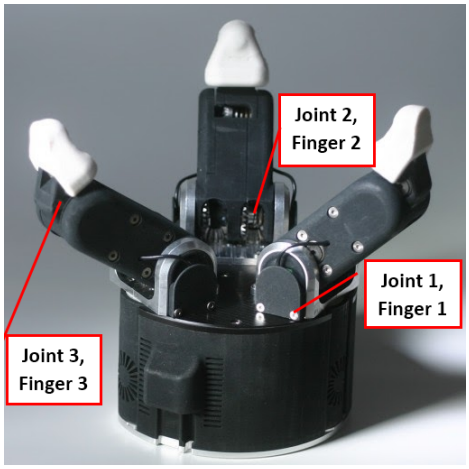

Figure 4.3: Robot hand with three fingers and three joints in each finger, Shadow Robot Company [1, 2].

joint, features including velocity, effort (torque) and position were measured. The dataset has 992,641 records of grasps and 28 features, with 448,046 records (45% of data), labeled as Stable grasps and 544,595 records (55% of data), labeled as Unstable grasps.

Following the recommendation in [1, 2], the joint position was excluded from the features. This is because the shape of the hand is object-specific, whereas our aim in this study is to predict the quality of the grasp in an object agnostic manner. As a result, 18 input features which measure effort (torque) and velocity of each joint in all of the three fingers, and a grasp quality feature (as the target label) were used in training our ML models. Table 4.1 provides more details about the features' notations. Fig. 4.3 also shows the robots' fingers and joints.

The data set, consisting of the above features and target, was randomly split into

[1]https://www.shadowrobot.com/shadow-smart-grasping-system/
[2]https://www.universal-robots.com/products/ur10-robot/

TABLE 4.2: ML Model Evaluation Metrics (Methodology 1)

Metric	Decision Tree	Logistic Regression	Tree Ensemble (LightGBM)	Neural Net (MLP)
Accuracy	0.79	0.74	0.83	0.82
AUC	0.84	0.81	0.91	0.90
F1	0.78	0.74	0.83	0.81
Precision	0.93	0.83	0.93	0.86
Recall	0.66	0.66	0.76	0.75

a training set (794,112 records) to learn the ML model, a validation set (99,265 records), and a test set (99,265 records) for evaluating and reporting the generalization ability of the model on unseen data.

4.1.2 Case Study 1

In the first study on grasp failure prediction, we chose the logistic regression (LR) [56] and Decision Tree (DT) classifiers as white box models and the LightGBM (gradient boosting classifier) as black box models to conduct our experiments. We then built the models using the training set and validated the results on the test set. Table 4.2 compares these four models based on the prediction metrics obtained by cross-validation. In the following, we report the results of our experiments in terms of the standard ML model's performance metrics of prediction accuracy (proportion of correct classifications), area under the ROC curve (AUC), precision (proportion of true positives out of the predicted positives), recall (proportion of true positives predicted relative to all the true positives), and F1 score (harmonic mean of precision and recall). All the metrics range in $[0, 1]$ with higher values indicating better performance.

As we can see in Table 4.2, LightGBM surpassed the DT and Logistic regression classifiers in all evaluation metrics.

In the following, we chose LightGBM for further prediction and explainability analysis, since it was the top performer in most metrics.

Fig. 4.4 shows LightGBM's prediction results on the test set. 41,674 of the records with the true success label were predicted correctly in the success class (True Negatives) and 41,028 of the true failures were predicted correctly to be a failure (True Positive).

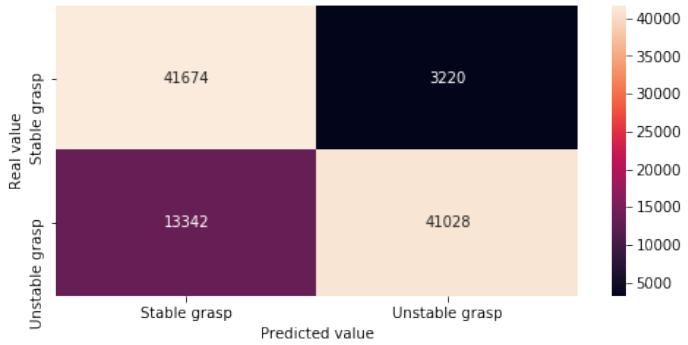

Figure 4.4: LightGBM confusion matrix

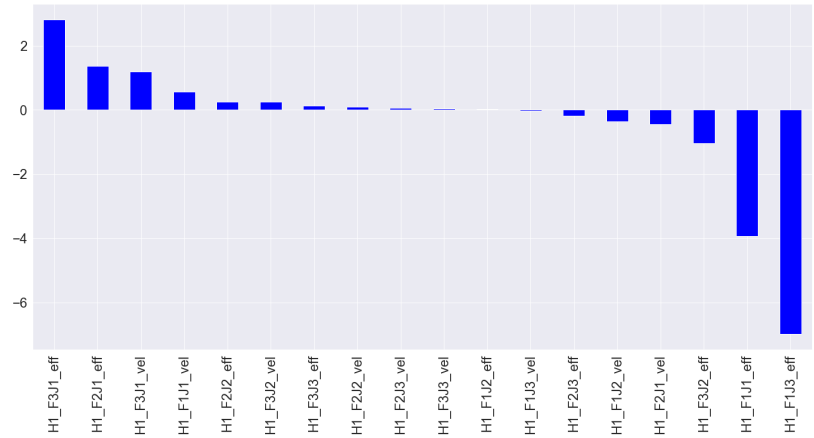

Figure 4.5: Global interpretability of the entire training data by the white-box Logistic Regression

4.1.2.1 Comparing White-Box and Black-Box Model Global Explanations

Important features in building a logistic regression model are achieved by calculating the coefficients of the features in the decision function. As shown in Figure 4.8, joint 1's effort (torque) in finger 2 and finger 3 and its velocity in finger 3 have positive contributions in predicting the likelihood of robot's grasp failure, whereas joint 1 and joint 3 effort (torque) in finger 1 have negative coefficients contributing most negatively. This means that joint 1's effort in finger 2 and finger 3 and its velocity in finger 3 make the failure more likely and joint 1 and joint 3's efforts in finger 1 make failure less likely. Based on the estimated coefficients, the effect of the rest of features on the prediction output is very small or zero.

Fig. 4.6 also shows the features, sorted according to their importance in making decisions in the DT model. It reveals that joint 1's effort (torque) in all fingers is more

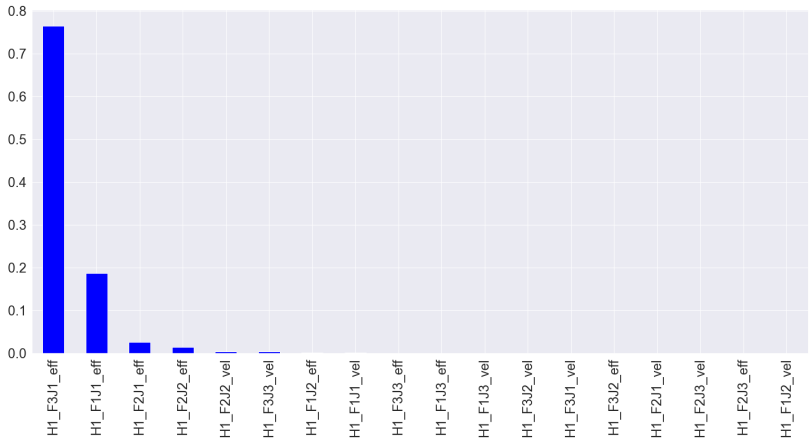

Figure 4.6: Global interpretability of the entire training data by the white-box DT classifier, joint 1 effort (torque) in all fingers is more important than velocity in predicting grasp stability

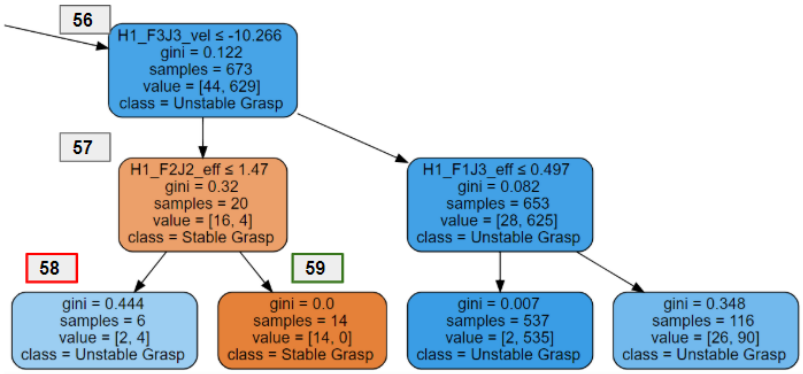

Figure 4.7: The DT classifier's decision paths for correct and incorrect classifications.

important than velocity in predicting grasp stability.

4.1.2.2 Comparing White-Box and Black-Box Model Local Explanations

Logistic regression does not provide local explanations for the individual data records while decision paths in the DT model show how the decisions were made. Figure 4.10, displays part of these decision paths for correct classifications and incorrect classification (misclassifications). For example, following nodes 0, 32, 48, 56, 57, 59, and paying attention to the feature values that led to the final decision, we can see that all 14 records were classified correctly in the class "Stable grasp"; while the path including nodes 0, 32, 48, 56, 57, 58 resulted in 4 incorrectly classified records and 2 correctly classified records.

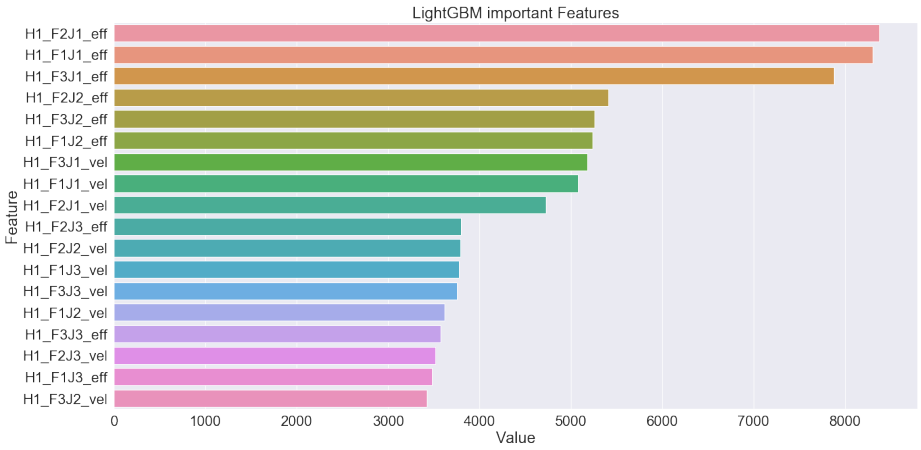

Figure 4.8: Global interpretability for the entire training data by the LightGBM black-box classifier based on feature importance.

4.1.2.3 Comparing Global and Local Explanations Generated by the SHAP Post-hoc Explanation Method

LightGBM classifier is a tree ensemble learning method. The output of the model is considered a black box because the model consists of many individual decision trees, which are built using randomly chosen variables, thus making it difficult for users and even experts to understand the decision process.

Although global "feature importance" has been used to interpret a LightGBM model, it only gives an overview of the contribution of the features in the prediction results for the entire training data (global interpretation) and not for individual samples. Also, feature importance is not considered to be a "consistent" approach, meaning that changing the model may decrease the importance of a feature even though the feature might still have a high impact on the model's output [71]. In contrast, another global interpretation method provided by SHAP has solved the consistency problem by using additive feature attribution methods and considering the attribution of the features in the output of the model. Moreover, SHAP provides global and local explanations for both training and test data sets.

According to Figure 4.11, Joint 1's and 2's effort (torque) in all fingers is more important than its velocity in predicting grasp stability (failure risk in the grasp).

In the following, we use the SHAP Tree explainer [71] on the prediction output of the

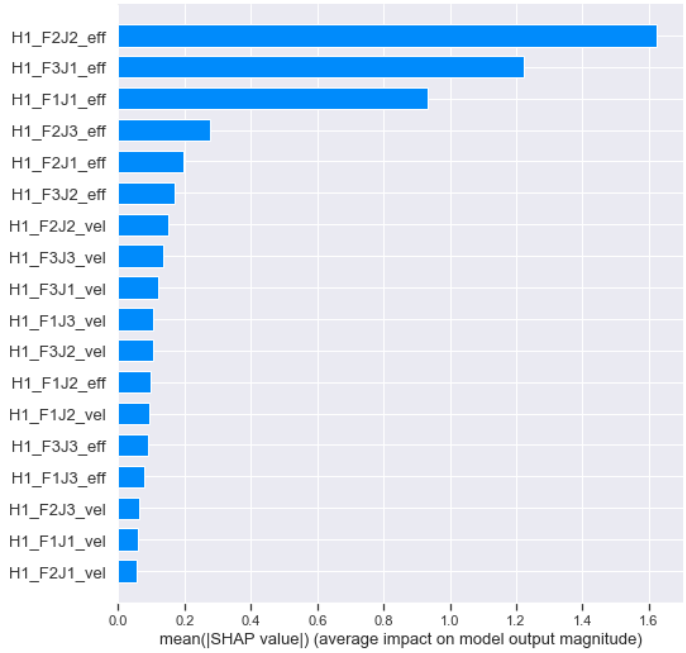

Figure 4.9: Global interpretability of the entire test set for the LightGBM model based on SHAP explanations

LightGBM model to see what information we can extract about the robot's grasp failure cases. As we mentioned, SHAP computes two types of explanations: Local and global explanations, as we will illustrate below.

To get a general view of which features are most important in failure prediction, we can check the global explanation plot for either training or test data set. This explanation ranks each feature based on its mean absolute Shapley value (global importance).

Figure 4.12 shows that based on the Shapley values among test samples, joint 2's effort in finger 2 is the most important feature in predicting grasp failure. Joint 1's effort in the other fingers is also important as shown in Figure 4.11.

To know how joint 2's finger 2 impacts the prediction of failure, we can examine Figure 4.13 which shows the distribution and value impact of the features in detail.

To display the explanation information in Figure 4.13, SHAP first sorts the features based on their global importance. Then dots, representing the SHAP values, are plotted horizontally. Each dot is colored by the value of that feature, from low (blue) to high (fuchsia).

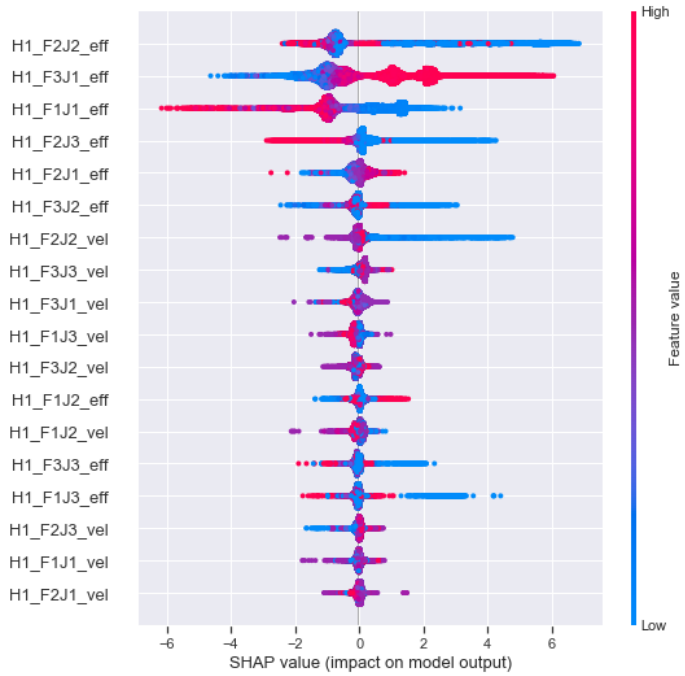

Figure 4.10: Distribution and value impact of the features of the test data for the LightGBM model, explained by SHAP Global interpretability

As we can see, lower values (blue dots towards the right) of joint 2's effort in finger 2 have higher impact on the model's output, whereas higher values (fuchsia dots), have lower impact. The next important feature is joint 1's effort in finger 3 which increases the risk of failures in the robot's grasp.

SHAP also provides explanations for any given data record (local explanation). To do that, SHAP decomposes the prediction in a graph and visualizes the feature's contribution to the prediction result. We chose one instance of true positive records from the test set and show its explanation in Figure 4.14. Each feature value is a force that either increases (positive values in fuchsia) or decreases (negative values in blue) the prediction of the failure.

Our model has predicted that the robot will fail in grasping this sample of data with probability almost 1 (output value in Figure 4.14). Also, the average of all predicted probabilities for the failure class in the test data (base value) is equal to 0.9038, which is the output value while ignoring all the input features.

Figure 4.11: Local explanations for a true positive case predicted to be in the Failure class by the LightGBM model

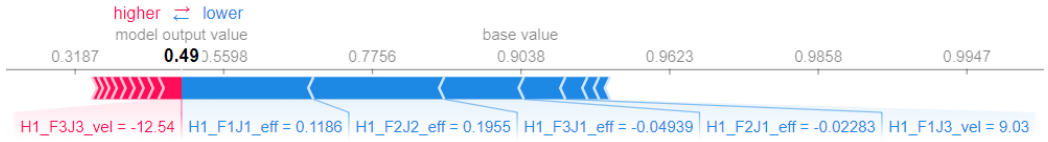

Figure 4.12: Local explanations for a false negative case among the prediction results of the LightGBM model.

According to Figure 4.14, features in fuchsia such as joint 2's effort in finger 2 (the most important features from global explanations) and joint 1's effort in finger 1 and 3 contributed to push the model's output from the base value that ignores all features (0.9038) toward the model's actual output that takes into account the features (probability of failure for this specific record which is equal to 1).

Figure 4.15 shows how the effort in joints' 1 finger 1 and joints' 2 finger 2 led the model to misclassify this sample of data as a success.

Figure 4.16 shows that while features in blue, such as joint 2's effort in finger 2 help the prediction to be correct (decreasing the probability of failure), the effects of the fuchsia features such as joint 1's and 3's effort in finger 2 led to incorrectly classify this sample.

With these predictions and explanations, an alternative planning of the task can be achieved through checking the reliability of other intervention methods. For example, this can be done by utilizing a variable autonomy based framework, where the level of autonomy

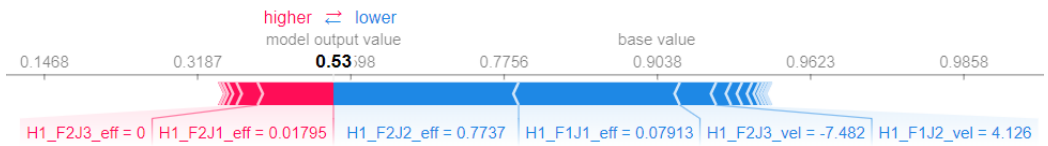

Figure 4.13: Local explanations for a false positive case for the LightGBM model prediction.

TABLE 4.3: ML model evaluation metrics (white-box vs glass-box)

Metric	Logistic Regression	Explainable Boosting Machine (EBM)
Accuracy	0.74	0.80
AUC	0.81	0.86
F1	0.74	0.78
Precision	0.83	0.93
Recall	0.66	0.68

can be chosen for specific sub tasks such that the overall performance is optimized. In case of a prediction of a failure during a task, by observing the explanation, either the level of autonomy can be changed to manual for human intervention or an alternate path planning process can be initiated for autonomous intervention for the fault.

Our experimental results and their analysis, illustrate the need for an accurate explanation of a predicted fault in order to be able to think about and choose an effective corrective measure.

The white-box model was the Logistic Regression classifier [56]. Explainable Boosting Machine (EBM) [9–11], was used as the glass-box model.

In our experiments, we use interpretML [11], an open-source Python package, for all the implementations. Following the recommendation in [11] the default parameters for the EBM were chosen to obtain a faster performance.

Table 4.5 compares the prediction metrics obtained by cross-validation of the ML models.

4.1.3 Case Study 2

As shown in Table 4.3, EBM surpassed the Logistic Regression (LR) classifier on all evaluation metrics. Fig. 4.14 and Fig. 4.15 display the ROC curves for LR and EBM, respectively. EBM has a higher AUC score (0.86) than the LR (0.81).

4.1.3.1 Comparing White-Box and Glass-Box Models' Global Interpretability

Both EBM and LR can measure the contribution of each feature in shaping the final decision made by the models. Fig. 4.16 and Fig. 4.17 show the importance of each feature

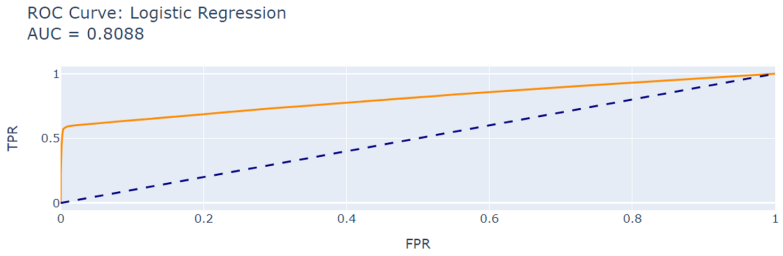

Figure 4.14: ROC curve (in orange) of the Logistic Regression Classifier. The y and x-axis are True Positive Rate and False Positive Rate, respectively. The blue line is the random guess which is the minimum performance benchmark.

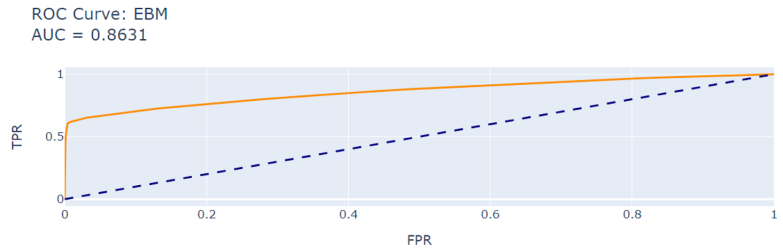

Figure 4.15: ROC curve (in orange), EBM. The y and x-axis are True Positive Rate and False Positive Rate, respectively. The blue line is the random guess which is the minimum performance benchmark.

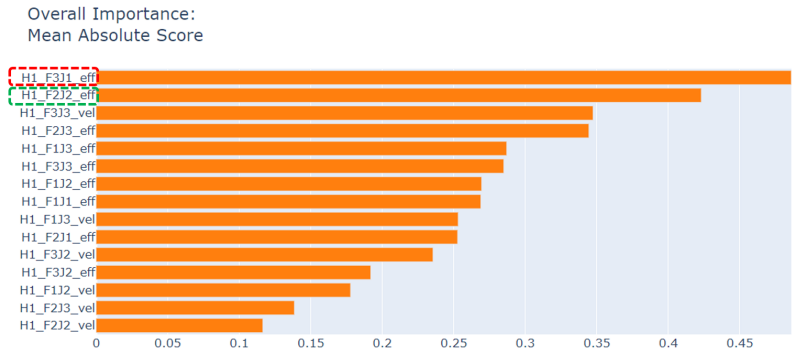

Figure 4.16: EBM Global Interpretability. Effort in joint 1 in finger 3 (outlined in red) and joint 2 in finger 2 (outlined in green) have globally important roles in predicting the risk of grasp failure.

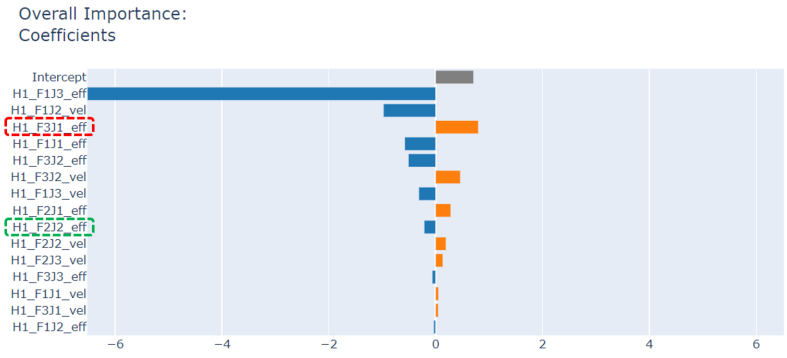

Figure 4.17: Logistic Regression Global Interpretability. Joint 1's effort in finger 3 (outlined in red), which is the most important feature in Figure 4.19, has the highest positive effect on the LR's output, as well. On the other hand, joint 2's effort in finger 2 (outlined in green), which is the second most important feature in the EBM model, has a very small negative impact on predicting the risk of failure of the grasp.

(globally) on the model's outcome over the training set.

Fig. 4.16 shows the global importance of features in building the EBM model. Effort in joint 1 in finger 3 (outlined in red) and joint 2 in finger 2 (outlined in green) have globally important roles in predicting the risk of grasp failure. Among the top 10 features which are ranked based on their overall importance, eight features are measuring effort and only two of them are velocity (joint 3's velocity in finger 3 and finger 1). This shows that applying proper effort to the object is more important in failing a grasp than velocity.

Fig. 4.17 displays the importance of features based on their coefficients (weights) in building the LR model. Features with positive contributions are presented in orange and features with negative effects are shown in blue. Comparing the feature importance plots of both models, it is interesting to note that joint 1's effort in finger 3, which is the most important feature in Figure 4.19, has the highest positive effect on the LR's output, as well. Meanwhile, joint 2's effort in finger 2, which is the second most important feature in the EBM model, has a very small negative impact on predicting the risk of failure of the grasp. It means increasing the effort in joint 2 of finger 2, decreases the risk of failure predicted by the LR model.

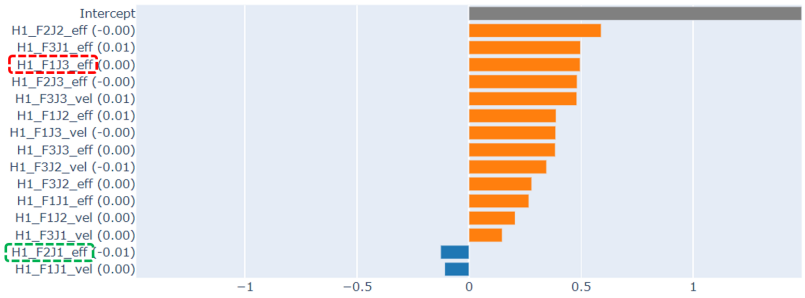

Figure 4.18: EBM's Local Interpretability. Feature joint 3's effort in finger 1 (outlined in red) has positive impact in predicting the risk of failure in the grasp. While joint 1's effort in finger 2 has negative effect (outlined in green).

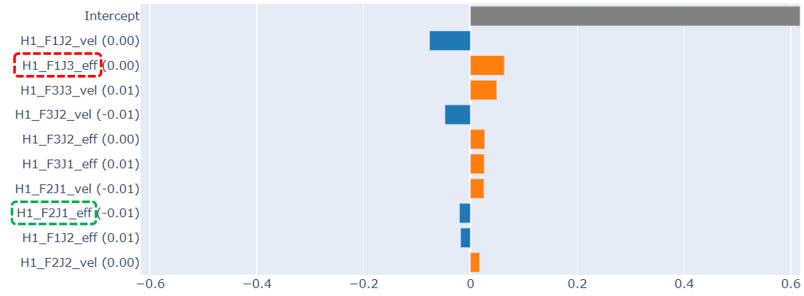

Figure 4.19: Logistic Regression's Local Interpretability. Feature joint 3's effort in finger 1 (outlined in red) has positive impact in predicting the risk of failure in the grasp. While joint 1's effort in finger 2 has negative effect (outlined in green).

4.1.3.2 Comparing White-Box and Glass-Box Models' Local Interpretability

The LR model provides only global interpretability. Thus, we used the LIME [63] technique to extract the local explanation for LR. In contrast, the glass-box algorithm (EBM) explains its own predictions both locally and globally.

To understand why and how both models made a decision (in this example a correct decision) about one of the test set samples, we chose one instance from the true positive records (a true failures which was predicted correctly to be a failure).

Fig. 4.18 and Fig. 4.19 show EBM and LR's individual explanations (local interpretability), respectively. The figures display how each feature has contributed to predict this grasp correctly in the Failure class. Features with positive effects are shown in orange and features with negative effects are shown in blue. Regardless of the features' order, both

TABLE 4.4: Prediction and explanation run-time (in seconds) and the figure showing the corresponding explanation in parentheses. While the white-box model (LR) is faster at prediction time than the glass-box model (EBM), LR is only globally interpretable and is not locally explainable. To obtain a local explanation, it takes almost 2 orders of magnitude longer for LR (which relies on LIME for the explanation generation) compared to a local explanation generated by EBM for its own prediction.

Prediction per instance (sec.)		Local explanation per instance (sec.)	
LR	EBM	LIME to locally explain LR	EBM
1e−07	1.7e−06	7.245187 (Fig. 4.18)	0.0289855 (Fig. 4.19)

local explanations have common features. For example, joint 3's effort in finger 1 (outlined in red) has positive impact in both plots. Also, both explanations agree on the negative effect of joint 1's effort in finger 2 in predicting the risk of failure in the grasp. Also, both plots show a rounded value for each feature following the feature's name; but because their values are very small, they do not seem very different.

Moreover, Table 4.4 shows that EBM performs significantly faster (0.03 seconds) than LIME (7.25 seconds) when it comes to generating and visualizing the local explanations, as shown in Fig. 4.18 and Fig. 4.19. This is due to the fact that unlike EBM, LIME must learn a surrogate model for each new test prediction case, which involves sampling enough training points in the test case's neighborhood, then training a surrogate model.

While the white-box model (LR) is faster at prediction time than the glass-box model (EBM), LR is only globally interpretable and is not locally explainable. To obtain a local explanation, it takes almost 2 orders of magnitude longer for LR (which relies on LIME for the explanation generation) compared to a local explanation generated by EBM for its own prediction.

All the experiments were conducted on a 2.20 GHz Intel Core i7 PC with 8 GB of RAM.

4.1.3.3 Comparing White-Box and Glass-Box Models' Shape Plots

One of the features provided by any GAM-based model is the ability to visualize the shape function of each feature in the data. This capability helps to understand how

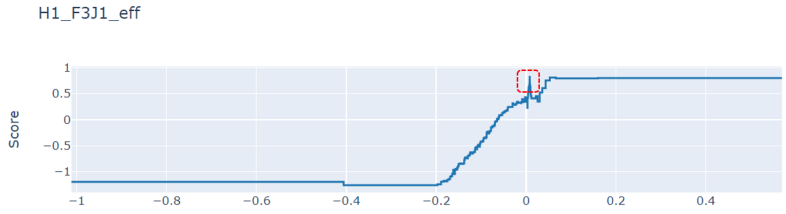

Figure 4.20: EBM's shape plots of joint 1's effort in finger 3. The risk of failing a grasp by the robot increases by 0.82 for the feature values between -0.2 and 0.05 (Nm), but it shows a high jump to a risk score of 0.82 when the effort is equal to 0.0083 (Nm).The x-axis displays the applied effort in joint 1's effort in finger 3 (Nm) and the y-axis shows the corresponding risk score of failure.

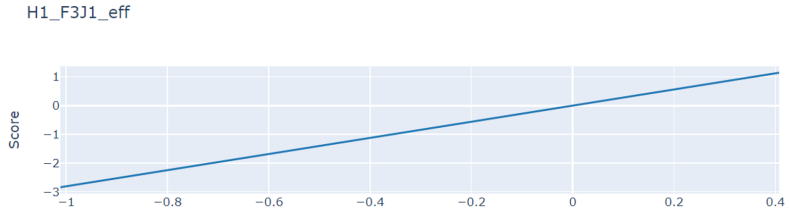

Figure 4.21: Logistic Regression's shape plots of joint 1's effort in finger 3 fails to learn and uncover any details about the feature behaviour in the learning process. The x-axis displays the applied effort in joint 1's effort in finger 3 (Nm) and the y-axis shows the corresponding risk score of failure.

each feature shapes the predicted results (in our case, predicting the risk of grasp failures). Moreover, it helps to capture any unexpected impact or even discover a meaningful result, as was done in healthcare research [10].

Fig. 4.20 shows the EBM's shape function for joint 1's effort in finger 3. According to the plot, the risk of failing a grasp increases by 0.82 for the feature values between -0.2 and 0.05 (Nm) but it shows a high jump in the risk score of 0.82 when the effort is equal to 0.0083 (Nm). For the rest of the features' values, the score does not have any noticeable change. In contrast, Fig. 4.21 shows that because the LR model is not able to capture non-linear relationships in the data, it fails to learn and uncover those details from joint 1's effort in finger 3.

4.2 A Methodology for Rank Similarity Analysis of Post-hoc Explanations Methods in Robotic Failure Prediction

In the third grasp failure prediction study, we investigate the following research questions:

- RQ3: Do the explanation methods agree on selecting the most responsible feature for grasp failures?
- RQ4: How similar are their results on ranking important features and their contributions in explaining the failures?
- RQ5: Do their rankings of important features depend on the type of predictive ML Model?
- RQ6: How to evaluate the explanation rankings?

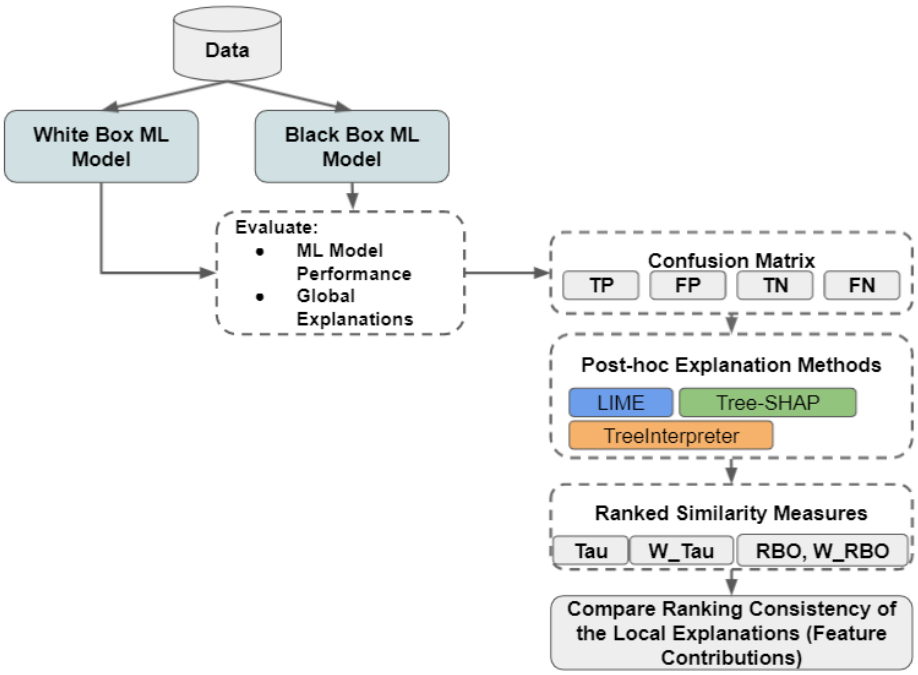

Figure 4.22: Methodology Flow Diagram 3, Rank Similarity Analysis of Post-hoc Explanation Methods

To explore these questions, we train and compare two different models on grasp failure simulation data, a white-box model (Decision Tree classifier [55]) which is inherently

interpretable and a black-box model (Random Forest classifier [51]) with high accuracy and no explanations about the prediction results. To explain individual predictions, we use TreeSHAP [71], LIME [63] and TreeInterpreter (TI) [66,67] explanation methods to provide local explanations for any sample of data. Fig. 4.22 depicts the flow of our methodology, starting with learning predictive models for failure, then generating explanations for predicted failures. After evaluating the prediction results of both white-box and black-box models, we apply the explanation methods to generate local explanations for each record in sub-samples of the test set (based on classification's output: False Negative (FN), True Negative (TN), False Positive (FP) and True Positive (TP)). Then, we use four ranking similarity metrics (Kendal Tau [78], Weighted Kendal Tau [84], RBO and Weighted RBO [79]) to measure the similarity between top-3 feature contributions generated by each explanation methods. Finally, by calculating the median of the correlation metrics in each sub-sample, we compare the consistency of the feature contributions. The experimental results are presented in Section 4.2.1.

4.2.1 Experimental Results for Rank Similarity Analysis of Post-hoc Explanations Methods in Robotic Failure Prediction

4.2.1.1 Case Study 3

In our third study on grasp failure prediction, we used the same data set and applied the same data preprocessing as we did in [85]. We tuned the hyperparameters of the algorithms using grid search, while evaluating the AUC of the model on the validation set.

Table 4.5 compares the prediction accuracy metrics obtained by cross-validation of the ML models. All the reported results are obtained from the average of a 5-fold cross-validation and consist of standard ML models performance evaluation metrics. The metrics all range in [0,1] with 1 being the best performance. Based on the mean and standard deviation (in parenthesis) of the results, we conclude that RF surpassed the DT in all evaluation metrics, as highlighted in the table. Since RF is an ensemble ML method which aggregates multiple DTs, it has higher accuracy than a single DT model.

TABLE 4.5: Mean (and standard deviation) of ML model performance obtained from the average of a 5-fold cross-validation, Random Forest (Black-box) vs Decision Tree (White-box). The best metric's value for each model, here based on the AUC score, is in bold.

Metric	Random Forest	Decision Tree
Accuracy	0.8020(+/ − 0.0001)	0.7947 (+/- 0.0030)
F1	0.7879 (+/- 0.0004)	0.7816 (+/- 0.0062)
Precision	0.9530 (+/- 0.0015)	0.9369 (+/- 0.0124)
Recall	0.6715 (+/- 0.0013)	0.6707 (+/- 0.0151)
AUC	**0.8712 (+/- 0.0002)**	**0.8524 (+/- 0.0053)**

Comparing White-Box and Black-Box Model Global Explanations To answer research question RQ5 from the global explanation perspective, we compared the global explanations from the decision tree and random forest classifiers.

As described in Section 2.3.6, the global explanation shows the importance of each feature (globally) on the model's outcome over the entire training data. The explanations are based on the Feature Importance (FI) score that measures the impact of each feature on predicting the output of the ML models. Therefore, we measured the (FI) scores[1] to study if applying different predictive ML models, in our case the decision tree and random forest classifiers, causes any changes (increase/decrease) in the feature importance (FI) score of a specific feature. Since both models were built using the Gini index criteria for node splitting, the FI scores were based on the Gini index, too.

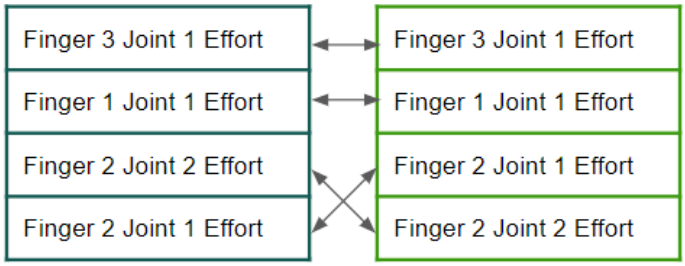

Figure 4.23: Comparison of the top-4 features ranked by Random Forest (left) and Decision Tree Classifier (right) based on Global Interpretability.

Fig. 4.23 shows the differences between the positions of the top-4 features which were ranked by the Random Forest (RF) (left) and Decision Tree (DT) (right) classifiers based

[1]We used the *feature-importance* function from the Scikit-learn library [74].

on the FI score. Comparing the ranked positions in the lists, we notice that the effort in the first joints of both finger 3 and finger 1, has the same importance on failing a grasp (predicted by the RF and DT models). Whereas, the effort in the first joint of finger 2 has different ranked positions (third position in the ranked-list by RF and fourth position in the ranked-list by DT). Also, the effort in the second joint of finger 2 has different positions. These differences show that the RF and DT models disagreed on the impacts that these two features could have on stability of the grasp and causing the failures.

The rest of the experimental results for Case Study 3 are presented in the following sections.

4.2.1.2 Comparing White-Box and Black-Box Models' Local Explanations

To answer research questions RQ3 and RQ4, we compared the local explanations generated after using the decision tree and random forest classifiers as the predictive models. The explanations were generated by three post-hoc explanation methods including LIME, Tree-SHAP and TI.

Fig. 4.24 and 4.25 show individual explanations (local explanations) for one of the true positive test samples, generated by LIME (in blue), Tree-SHAP (in green) and TI (in orange) from the predicted outputs of DT and RF model, respectively. To compare the local explanations, we chose one instance from the true positive records (a true failure which was predicted correctly to be a failure) from both ML models (DT and RF). The figures display how each feature has contributed to predict this grasp correctly in the **Failure** class. The figure also displays if the features have positive effects (with contributions to the right direction) and if they have negative effects (with contributions to the left direction). Fig. 4.24 shows that based on the Tree-SHAP explanations (in green) and TI (in orange), joint 1's effort in finger 3 is the most responsible feature in grasping failure. While LIME (in blue) selected the joint 1's effort in finger 1 as the most important feature in failing a grasp.

Comparing the RF individual (local) feature contributions, for the same true positive

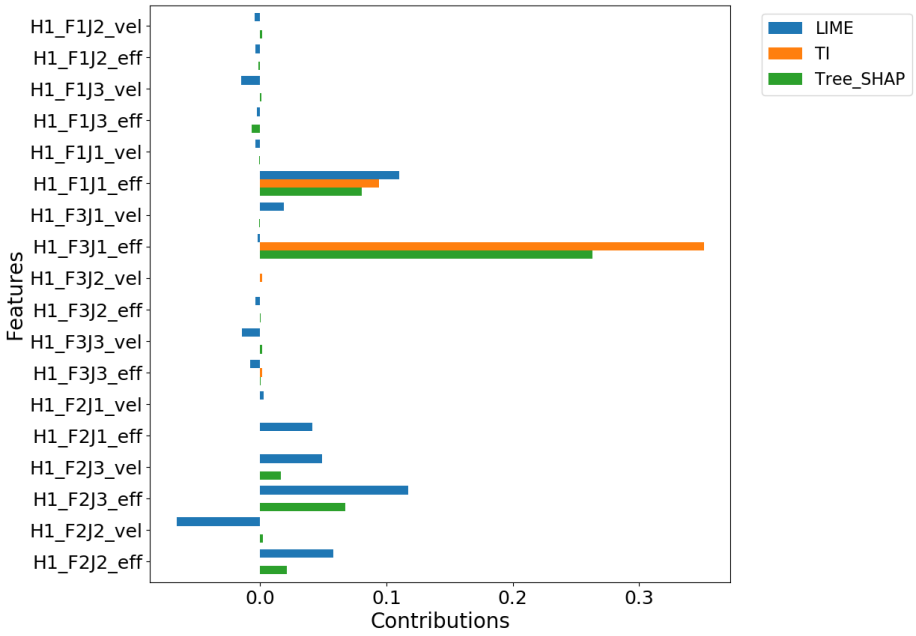

Figure 4.24: Decision Tree individual (local) feature contributions for one of the True Positives. According to Tree-SHAP explanations (in green) and TI (in orange), joint 1's effort in finger 3 is the most responsible features in grasping failure. While LIME (in blue) selected the joint 1's effort in finger 1 as the most important features in failing a grasp.

sample, reveals that based on the Tree-SHAP explanations (in green), joint 1's effort in finger 1 and finger 3 are the most responsible features in grasping failure. While TI (in orange) selected joint 1's effort in finger 3, and LIME (in blue) found the joint 1's effort in finger 1, as the most important features in failing a grasp. In this example, all the explanation methods mostly agree on the importance of joint 1's effort in finger 1 and finger 3 in predicting the instability of the grasp and causing failure.

Moreover, Table 4.6 compares the implementation run-time of the explanation methods for both DT and RF models and for each instance of the test set. All the experiments were conducted on a 2.20 GHz Intel Core i7 PC with 8 GB of RAM and implementing on Google Colab (cloud). The results show that TreeInterpreter (TI) runs one order of magnitude faster than other Tree-SHAP and 5 orders of magniture faster than LIME to generate local explanations in both ML models as shown in Fig. 4.26. This is due to the fact that unlike the Tree-SHAP and *TI*, *LIME* must learn a surrogate model for each new test prediction case, which involves sampling enough training points in the test case's neighborhood.

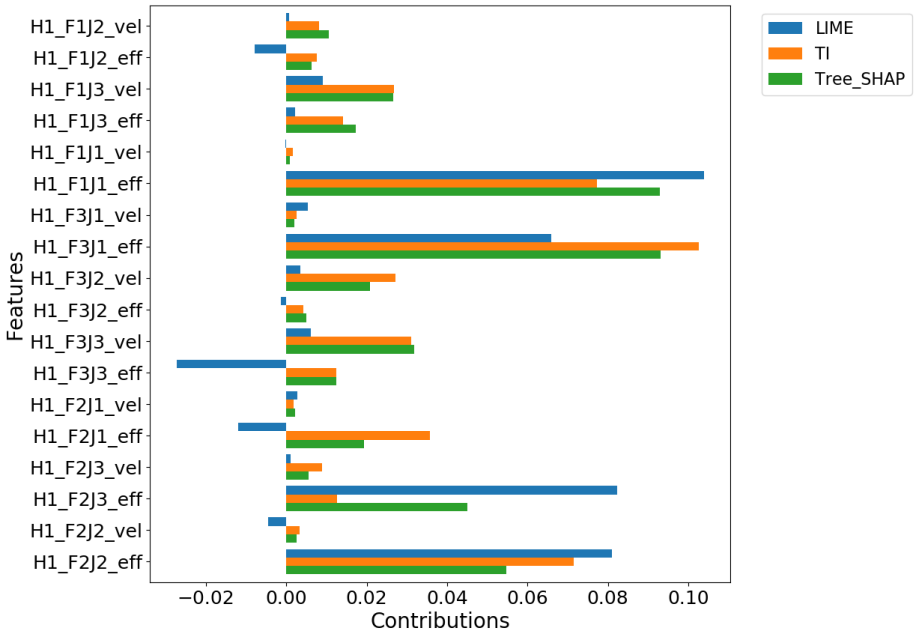

Figure 4.25: Random Forest individual (local) feature contribution for one of the True Positive instances. According to Tree-SHAP explanations (in green), joint 1's effort in finger 1 and finger 3 are the most responsible features in grasping failure. While TI (in orange) selected joint 1's effort in finger, and LIME (in blue) found the joint 1's effort in finger 1, as the most important features in failing a grasp.

TABLE 4.6: Comparing the average run-time in seconds (per instance) of three Post-hoc Explanations Methods (Tree-SHAP, Treeinterpretr and LIME) between two ML Models (Decision Tree and Random Forest Classifier).

ML Model	Tree-SHAP [1]	TreeInterpreter [2]	LIME [3]
Decision Tree	0.00024	0.00001	3.33447
Random Forest	0.07646	0.00234	4.66648

Thus, although both TI and Tree-SHAP generate explanations for tree-based models, in our study TI shows faster performance than Tree-SHAP. TI generates the feature contributions based on the series of decisions in a tree (from the root of the tree to the leaves for each instance of data) and it only considers one order of the features. Whereas, Tree-SHAP calculates the average of all possible orders among the features. For this reason, TI has a faster running time than Tree-SHAP. This also shows that TI has more transparency than Tree-SHAP.

In addition, the comparison of the performance of all the post-hoc explanation meth-

ods in generating explanations from DT and RF models reveals that in general, the explanation methods have faster run-time in generating explanations from DT compared to RF. This is not a surprising result as the DT has a less complex model than RF (see Table 4.6).

Considering one true positive sample, our initial experiments showed the inconsistency and differences in the feature contributions generated by Tree-SHAP, Treeinterpretr and LIME. Therefore to conduct our experiments, we applied the post-hoc explanation methods to generate local explanations for each record from the sub-samples of the test set, after partitioning the test data instances based on the classification model's output, namely including False Negatives (FN), True Negatives (TN), False Positives (FP), and True Positives (TP).

Next, we used four ranking similarity metrics, Kendal Tau [78], Weighted Kendal Tau [84], RBO, and Weighted RBO [79], to measure the similarity between the top-3 feature contributions generated by each explanation method and to answer research question RQ6.

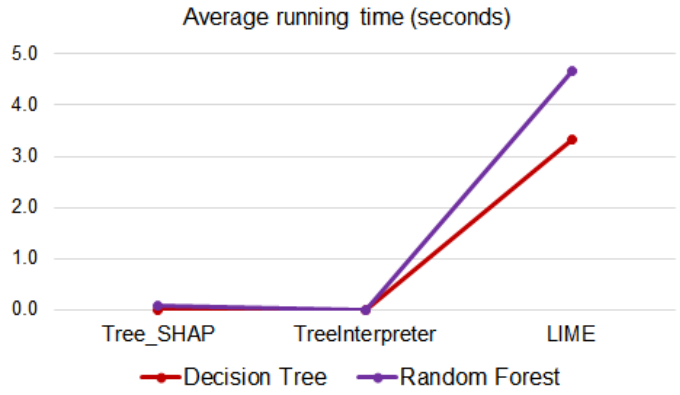

Figure 4.26: Average of Post-hoc Explanations Methods Run-time Per Instance (seconds).

We used Kendall's Tau_b[4] to assess the consistency of the explanations. In the weighted version of Kendall's Tau[5], the exchange of items with higher weights has more

[1]https://github.com/slundberg/shap
[2]https://github.com/andosa/treeinterpreter
[3]https://github.com/marcotcr/lime
[4]We applied the Kendall's Tau_b with the default values for all the input parameters. To apply Kendall's Tau and Weighted Kendall's Tau, we used two implementations provided by Scipy statistical module. The Kendall's Tau implementation by default calculates a version of the original Kendall's Tau (Kendall's Tau_b) which handles ties between ranked items (when two items have the same rank): https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.kendalltau.htmlscipy.stats.kendalltau

impact than the exchange of items with lower weights. Therefore, to assign more weights to the top-n items/features by ranking the items based on their indices and assigning a weight of $\frac{1}{1+r}$ to each item with rank r.

To compare the effect of the weighting used by RBO on the top-3 feature contributions generated by each post-hoc explanation method, we used $p = 0.5$ for Weighted-RBO and $p = 1$ for RBO. As mentioned in Sec. 2.3.7.2, having $p = 1$ makes the weights of each ranked feature contribution the same, as with the default Kendall's Tau, in which the rank or position does not matter.

Finally, by calculating the median of the rank similarity metrics in each sub-sample, we compared the consistency of the feature contributions. In evaluating the explanations, we used the absolute value of the feature contributions. Reviewing LIME's implementation[1], we confirmed that this method provides the explanations of the features with their indices as a dictionary, including a sorted list of the explanations based on their absolute values. In our experiments, we thus followed the same strategy and we used the original indices generated by LIME to order and extract the feature contributions. Similarly, for Tree-SHAP and TreeInterpreter, we sorted the explanations based on their absolute values. Then we extracted the index of features which were sorted based on their absolute values. Applying the same strategy in extracting the outputs of the post-hoc explanation methods helped to make the comparison between all the methods consistent.

The results in Table 4.7 summarize the ranking similarity of the explanations (feature contributions) between each pair of post-hoc explanation methods for the decision tree classifier outputs. The greater values (shown in bold) indicate that the explanations were more similar based on their ranking positions. Considering the Kendall Tau and Weighted Kendall Tau metrics, in the True Positive sub-sample, all the explanation methods show very strong agreement (median = +1) on the ranking of the feature contributions. In other

[5]We used the Weighted Kendall's Tau implementation from https://docs.scipy.org/doc/scipy/reference/generated/scipy.stat
We set the parameter *rank* as False to rank the items based on their indices directly and we use parameter *weigher*, which by default assigns a weight of $\frac{1}{1+r}$ to each item with rank r.

[6]we used one implementation for both RBO and Weighted RBO metrics but with two different values for the input parameter p: https://github.com/changyaochen/rbo

words, all the ranked lists generated by each pair of explanation methods are identical. The agreement between Tree-SHAP and TI is lower for the rest of sub-samples (with the median of +0.3 or +0.2). The consistency between each pair of explanation methods is low (with median of -0.3, -0.2 or +0.3) for the other sub-samples. For the True Negatives and False Negative sub-samples, all the post-hoc methods had the same level of agreement (correlation) on determining the feature contributions.

In contrast, RBO and Weighted RBO captured more similarity between *Tree-SHAP* and TI compared to the rest of the methods, the ranked lists of feature contributions generated by LIME and TI had no similarities (median = 0). Recall that Table 2.1 provided more details about the range of values for each rank similarity metric. The table also defines how each value indicates similarity or dissimilarity between the ranked lists. For example, both RBO and Weighted RBO have a range of [0,1], while Kendall Tau and Weighted Kendall Tau have a range of [-1,1]. The value of -1 in the Kendall Tau and Weighted Kendall Tau shows that the similarity (correlation) is strong but features (in the ranked lists) are in the reverse orders, which means the lists are very dissimilar.

Fig. 4.27 to 4.30 visualize the same information as shown in Table 4.7. A summary of the ranking similarity analysis on Random Forest's sub-samples is presented in Table 4.8. Considering the Kendall Tau and Weighted Kendall Tau metrics, for the False Positive and True Positive sub-samples, Tree-SHAP and TI show very strong agreement/consistency (median = +1) on the ranking of the feature contributions. In contrast, these two explanation methods have low consistency for the True Negative samples (median = +0.3) and moderate consistency for the False Negative sub-samples (median = +0.5). Agreement is defined if the correlations are positive and disagreement is defined if the correlations are negative.

Using the RBO and Weighted RBO metrics and assigning more weights to the features at the top of the lists, we notice that the consistency between Tree-SHAP and TI explanations, on all sub-samples, has increased (median = +0.8 or +0.9). Thus these two metrics captured more similarity between Tree-SHAP and TI compared to the rest of the

TABLE 4.7: Comparing the different explanations generated from the Decision Tree classifier's outputs based on ranking similarity (median). Note that RBO and Weighted RBO have a range of [0,1] while Kendall Tau and Weighted Kendall Tau have a range of [-1,1]. Bold means

(a) True Negatives

Explanation Method Pair	Kendall Tau	Weighted Kendall Tau	RBO	Weighted RBO
Tree-SHAP & TI	**0.3**	**0.2**	**0.7**	**0.7**
Tree-SHAP & LIME	**0.3**	**0.2**	0.3	0.2
LIME & TI	**0.3**	**0.2**	0.0	0.0

(b) False Negatives

Explanation Method Pair	Kendall Tau	Weighted Kendall Tau	RBO	Weighted RBO
Tree-SHAP & TI	**0.3**	**0.2**	**0.7**	**0.7**
Tree-SHAP & LIME	**0.3**	**0.2**	0.3	0.2
LIME & TI	**0.3**	**0.2**	0.0	0.0

(c) True Positives

Explanation Method Pair	Kendall Tau	Weighted Kendall Tau	RBO	Weighted RBO
Tree-SHAP & TI	**1**	**1**	**0.9**	**0.8**
Tree-SHAP & LIME	**1**	**1**	0.4	0.2
LIME & TI	**1**	**1**	0.3	0.2

(d) False Positives

Explanation Method Pair	Kendall Tau	Weighted Kendall Tau	RBO	Weighted RBO
Tree-SHAP & TI	**0.3**	0.2	**0.6**	**0.7**
Tree-SHAP & LIME	**0.3**	**0.3**	0.4	0.2
LIME & TI	-0.3	-0.2	0.3	0.2

methods. In contrast, the ranked lists of feature contributions generated by LIME and TI had weak similarities (median = 0).

The heat maps in Fig. 4.31-4.34 visualize the same information as shown in Table 4.8. The agreement is defined if the correlations (similarities) are positive and disagreement is defined if the correlations are negative. In Fig. 4.27, weak positive correlations (similarities) are shown as small blue circles and the weak negative correlations are shown as small red circles. In the heat maps, different similarity metrics are shown with different shapes. Thus circle, squares, triangles and hexagons, display the similarity values for Kendall Tau, Weighted Kendall Tau, RBO and weighted RBO, respectively.

Considering the Decision Tree Classifier's outputs in Table 4.7, Kendall Tau and Weighted Kendall Tau show the same similarities between each pair of explanation methods. In other words, adding weights to Kendall Tau did not have any effect on capturing the

TABLE 4.8: Comparing Explanations for Random Forest Classifier outputs based on ranking similarity (median). Please note that RBO and Weighted RBO have a range of [0,1] while Kendall Tau and Weighted Kendall Tau have a range of [-1,1].

(a) True Negatives

	Kendall Tau	Weighted Kendall Tau	RBO	Weighted RBO
Tree-SHAP & TI	**0.3**	**0.5**	**0.8**	**0.8**
Tree-SHAP & LIME	**0.3**	0.2	0.6	0.3
LIME & TI	-0.3	-0.4	0.4	0.2

(b) False Negatives

	Kendall Tau	Weighted Kendall Tau	RBO	Weighted RBO
Tree-SHAP & TI	**0.3**	**0.5**	**0.9**	**0.8**
Tree-SHAP & LIME	**0.3**	0.2	0.6	0.3
LIME & TI	-0.3	-0.4	0.4	0.2

(c) True Positives

	Kendall Tau	Weighted Kendall Tau	RBO	Weighted RBO
Tree-SHAP & TI	**1**	**1**	**0.9**	**0.8**
Tree-SHAP & LIME	0.3	0.5	0.4	0.2
LIME & TI	0.3	0.5	0.4	0.2

(d) False Positives

	Kendall Tau	Weighted Kendall Tau	RBO	Weighted RBO
Tree-SHAP & TI	**1**	**1**	**0.9**	**0.8**
Tree-SHAP & LIME	-0.3	-0.4	0.3	0.2
LIME & TI	-0.3	-0.4	0.3	0.2

similarities between the explanation methods. In contrast, the results in Table 4.8 show that using Random Forest Classifier (which is a more accurate model than the Decision Tree classifier) has affected the Weighted Kendall Tau's ability to capture the similarity between each pair of explanation methods. Therefore, there is a higher consistency between the explanations generated by *Tree-SHAP* and *TI* from the RF classifier's outputs.

Comparing the RBO and Weighted RBO similarity results, from both the DT and RF's outputs, we observe that the similarity between the explanations generated by *Tree-SHAP* and *TI* is higher than the rest of the explanations methods.

Our experiments showed that applying proper effort to the object and particularly the effort of joint 1 in finger 3 (for which both models agreed on its global importance in predicting failures) plays an important role in the stability of the grasps. From the mechanical design, in a three-finger gripper, the effort (power) that the first joints (joint 1),

Figure 4.27: Similarities between the Top-3 ranking lists of feature contributions generated by the different post-hoc explanation methods and measured by Kendall Tau (based on median) for the Decision Tree Classifier's outputs. In the True Positive sub-sample, all explanation methods show very strong agreement/consistency (median = +1) on the ranking of the feature contributions. In other words, all the ranked lists generated by each pair of explanation methods are identical. The agreement between Tree-SHAP and TI is lower in the rest of sub-samples (median = +0.3). The consistency between each pair of explanation methods is low (either median = +0.3 or -0.3) in other sub-samples.

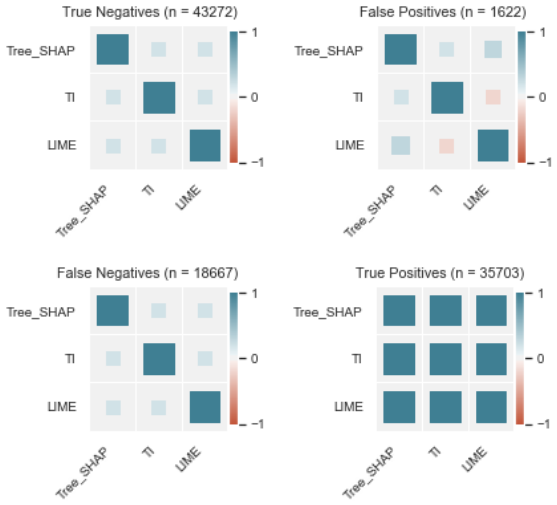

Figure 4.28: Similarities between the Top-3 ranking lists of feature contributions generated by different post-hoc explanation methods and measured by Weighted Kendall Tau (based on median) for the Decision Tree Classifier's outputs. In the True Positive sub-sample, all explanation methods show very strong agreement/consistency (median = +1) on the ranking of the feature contributions. In other words, all the ranked lists generated by each pair of explanation method are identical. The agreement between Tree-SHAP and TI is lower in the rest of sub-samples (median = +0.2). The consistency between each pair of explanation methods is lower in the rest of sub-samples (either median = +0.2 or -0.2). Same in the False positive sub-sample, the similarity between LIME and Tree-SHAP is low (median = +0.3).

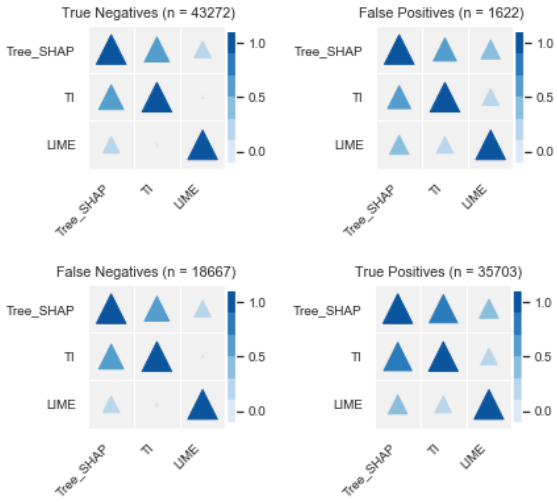

Figure 4.29: Similarities between the Top-3 ranking lists of feature contributions generated by different post-hoc explanation methods and measured by RBO (rank-biased overlap) for Decision Tree Classifier's outputs (based on median). In the True Positive sub-sample the consistency between Tree-SHAP and TI explanations is high (median = +0.9). While it is more moderate in the rest of sub-samples (median = +0.7 in the True Negatives and the False Negative and median = +0.6 in the False Positives). The median of 0 between explanations of LIME and TI in the False Negative and True Negative sub-samples shows that there is no similarity between the Top-3 feature contributions generated by these two methods. In other words, the ranked lists are disjoint.

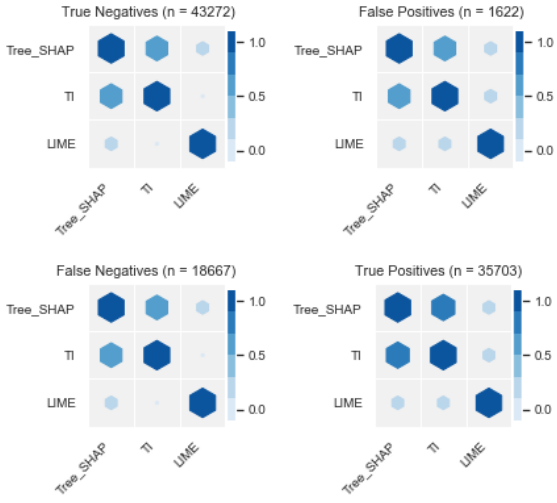

Figure 4.30: Similarities between the Top-3 ranking lists of feature contributions generated by different post-hoc explanation methods and measured by Weighted RBO (rank-biased overlap) for Decision Tree Classifier's outputs (based on median). Compared to the RBO results, the figure shows that assigning more weights to the top of the ranked lists does not change the consistency between each pair of explanations methods in the True Negative and False Negative sub-samples (the results of RBO and Weighted RBO is the same). The exceptions are in the True Positive and False Positive sub-samples where Weighted RBO was able to capture more similarity between Tree-SHAP and TI. In the False Negative and True Negative sub-samples, the similarity between Top-3 feature contributions generated by LIME and TI is 0. Thus, the ranked lists are disjoint.

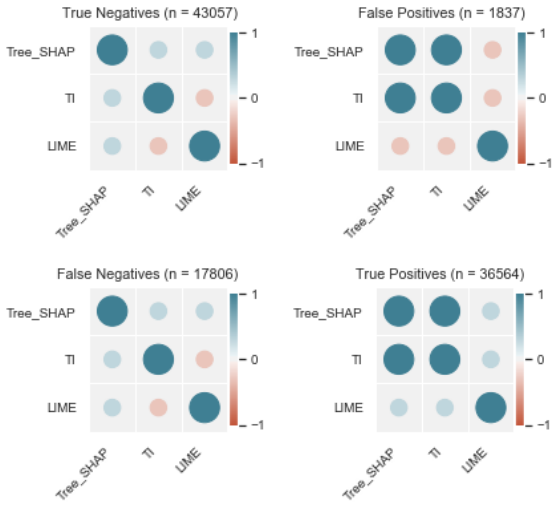

Figure 4.31: Similarities between the Top-3 ranking lists of feature contributions generated by different post-hoc explanation methods and measured by Kendall Tau (based on Median) for Random Forest Classifier's outputs. In the False Positive and True Positive sub-samples, Tree-SHAP and TI show very strong agreement/consistency (median = +1) on the ranking of the feature contributions. Whereas these two explanation methods have lower consistency in the True Negative (median = +0.3) and the False Negative (median = +0.3) sub-samples. Agreement is defined if the correlations (similarities) are positive and disagreement is defined if the correlations (similarities) are negative. Weak positive correlations are shown as small blue circles and weak negative correlations are shown as small red circles. The (dis)agreement between the rest of the explanation methods are not as strong as Tree-SHAP and TI.

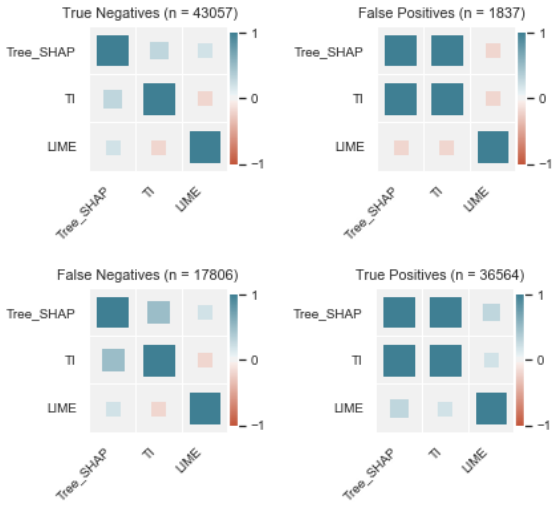

Figure 4.32: Similarities between the Top-3 ranking lists of feature contributions generated by different posts and measured by Weighted Kendall Tau (based on Median) for Random Forest Classifier's outputs. In the False Positive and True Positive sub-samples, Tree-SHAP and TI show very strong agreement/consistency (median = +1) on the ranking of the feature contributions. While these two explanation methods have lower consistency in the True Negative (median = +0.3) and moderate consistency in the False Negative sub-samples (median = +0.5).

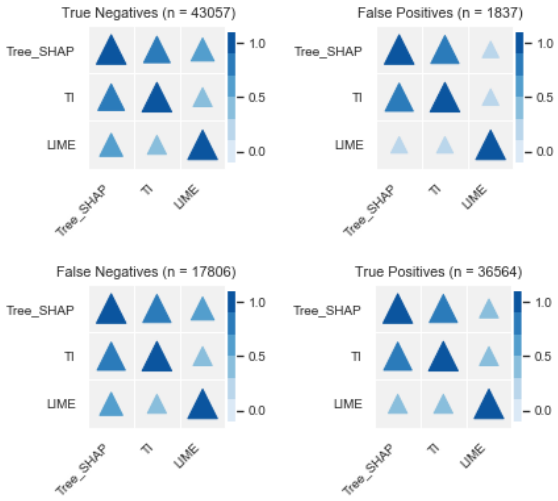

Figure 4.33: Similarities between the Top-3 ranking lists of feature contributions generated by different post-hoc explanation methods and measured by RBO (rank-biased overlap) for Random Forest Classifier's outputs. As shown in the figure, having common (similar) feature contributions' positions in the ranked list at different depth, increases the consistency between Tree-SHAP and TI explanations among all of the sub-samples. The similarities are strong in the False Positive and True Positive sub-samples (median = +0.9). Same in the False Negatives and True Negatives with a high similarity (median = +0.8).

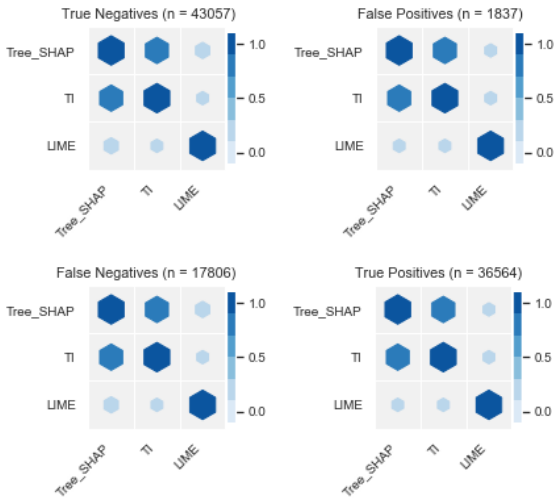

Figure 4.34: Similarities between the Top-3 ranking lists of feature contributions generated by different post-hoc explanation methods and measured by Weighted RBO (rank-biased overlap) for Random Forest Classifier's outputs. As shown in the figure, assigning more weight to the top of the ranked lists, increases the consistency between Tree-SHAP and TI explanations among all of the sub-samples (median = +0.8).

on all fingers, apply to make the first contact with the object, has an important impact on the failure or success of the grasping. Thus our results match with the mechanical concepts. In a collaborative task between a human operator and a robot, the prediction of failure, *especially when enriched with an explanation of the reasons leading to the failure*, can help the human user to take appropriate actions and prevent the potential risks.

Such information about a system is crucial during collaborative tasks involving a human operator and a robot. For a given task model, a prediction of failure is needed, in addition to providing an explanation for the prediction. The explainability can enable a quick and appropriate intervention by a human user in case of failure. Also, the failure prediction can allow the user to take preemptive measures to avoid the conditions.

4.3 Summary

In this chapter, we presented and evaluating three methodologies, presented through three case studies, for investigating the prediction of robotic grasp failures and for providing explanations to better understand the failure reasons in the grasping mechanism.

In the first case study, in Section 4.1.2, we compared the explainability and accuracy of the white-box models (which are inherently interpretable but less accurate) with black-box models (which do not provide any explanations but they have high accuracy). To get more insight about the robot hand failure reasons, we then compared global and local explanations generated by white-box and black-box models. To generate the local explanations from the predictions output of the selected BB model, we used the SHAP explanation method for tree-based models. We found that the explanations agree with concepts from the mechanical design of graspers. Some of our results have been published in [85].

In the second grasp failure prediction case study, in Section 4.1.3, we compared two different models on grasp failure simulation data, a white-box model (Logistic Regression [56]) which is inherently interpretable and a glass-box model (EBM) [9–11] which is designed to have a comparable accuracy to black-box models, while maintaining its interpretability. Our experiments in Subsection. 4.1.3.1 showed that applying proper effort to the object and

particularly to joint 1 in finger 3 (for which both models agreed on its global importance in predicting failures) plays an important role in the stability of the grasps. Furthermore, while the white-box model (LR) is faster at prediction time than the glass-box model (EBM), LR is only globally interpretable and is not locally explainable. To obtain a local explanation, it takes almost 2 orders of magnitude longer for LR (which relies on LIME for the explanation generation) compared to a local explanation generated by EBM for its own prediction.

We also extended our empirical analysis of robotic grasp failure prediction and explainability in a third case study (in Sec. 4.2.1.1) by applying three different post-hoc explanation methods, including *Tree-SHAP*, *LIME*, and *TreeInterpreter*. Then, we compared the similarity between the local explanations generated by these post-hoc explanation methods, using two different ranking similarity metrics, t*Kendall's Tau* which is a correlation based metric and *RBO* which is an intersection based metric that addresses the limitations of the original *Kendall's Tau* metric. The results show that the local explanations generated from the output of a black-box model (Random Forest, in our case) are more consistent in ranking of feature contributions than the local explanations generated from the output of a white-box model (a Decision Tree classifier in our study). This shows that the consistency of the explanations (rankings of important feature contributions in our case), generated from different explanation methods, depends on the type of predictive ML model used to make the predictions in the first place.

CHAPTER 5

CONCLUSION

In this work, we studied Machine Learning criteria that go beyond accuracy in two different problems where accuracy alone may not be sufficient as a performance goal: 1) in collaborative filtering recommendation where we study explainability and bias in addition to accuracy and 2) in robotic grasp failure prediction where we study explainability in addition to prediction accuracy. A summary of our contributions are listed as follows:

1. We proposed new recommendation algorithms which provide both debiasing and explainability simultaneously, called *Debiased Explainable Matrix Factorization*, to study the trade-offs between *accuracy*, *explainability*, and *bias*.

2. We studied the effectiveness of down-weighting popular items in rating prediction for unseen items and in recommending more explainable items to users.

3. We explored the trade-offs between prediction *accuracy* and *explainability* in another ML application, namely robotic grasp failure prediction by comparing the performance of white-box, black-box, and glass-box models.

4. We extended our study of explainable robotic grasp failure prediction by evaluating the consistency of local explanations generated by three different leading post-hoc explanation methods, including Tree-SHAP, LIME, and TreeInterpreter, in ranking feature contributions and finding the most responsible features for explaining predicted grasp failures.

Our experiments in Chapter 3 showed that adding explainability (from *MF* to *EMF*) and then debiasing the explainability regularization term (from *EMF* to *Debiased_EMF_2*)

has improved the accuracy and explainability of the recommended items, while reducing their novelty and diversity compared to the models with a debiasing term in their rating prediction loss such as *Popularity_Propensity_MF* [82], *Debiased_EMF_1*, and *Debiased_EMF_1_2*. These observations illustrate the trade-offs between accuracy, explainability, and popularity-debiasing in the performance of the models. Being able to assess the cost of debiasing on the model's accuracy and explainability can help researchers and data scientists to make better decisions about which models to deploy depending on the recommendation domain.

The limitations of the work on debiasing explainable recommendations include the need to conduct user studies to assess the impact on real systems, involving actual users.

In the second part of our work, we focused on an application of ML in robotic failure prediction. Robots that perform high risk tasks can benefit from predicting an impending failure. Furthermore, and especially in cases involving a robot collaborating with humans, there is a need for an explanation of failure predictions.

In our first case study in grasp failure prediction, we compared explainability and accuracy of the white-box models (which are inherently interpretable but less accurate) with black-box models (which do not provide any explanations but they have high accuracy). To get more insight about the robot hand failure reasons, we then compared global and local explanations generated by white-box and black-box models. To generate the local explanations from the predictions output of the selected BB model, we used SHAP explanation methods for tree-based models.

Our results showed that the explanations agree with concepts from the mechanical design of graspers. Although the 3-finger grasper examples were simple, the explainable machine learning approach promises to help uncover failure mechanism causes in new, less familiar systems.

In the second study, we found that while the white-box model (LR) is faster at prediction time than the glass-box model (EBM), LR is only globally interpretable and is not locally explainable. To obtain a local explanation, it takes almost 2 orders of magnitude

longer for LR (which relies on LIME for the explanation generation) compared to a local explanation generated by EBM for its own prediction.

We finally extended our study in robotic grasp failure prediction, in a third study, by applying and evaluating the consistency of three different post-hoc explanation methods, including Tree-SHAP, LIME, and TreeInterpreter. Then, we compared the local explanations generated by these leading post-hoc explanation methods, using different ranking similarity metrics, based on *Kendall's Tau*, which is a correlation based metric, and RBO, which is an intersection based metric, that solves the limitations of the original *Kendall's Tau* metric.

The results show that the local explanations generated from the output of a black-box model (Random Forest, in our case) are more consistent in ranking of feature contributions than the local explanations generated from the output of a white-box model (a Decision Tree classifier in our study). This shows that the consistency of the explanations (rankings of important feature contributions in our case), generated from different explanation methods, depends on the type of predictive ML model used to make the predictions in the first place.

While we were able to perform our robotic failure prediction study efficiently using an existing benchmark simulation dataset, future work needs to address real experimental data collection and actual task-based validation, since simulations are not a perfect representation of reality. This in turn must address the challenge of having to collect enough representative data from real experiments. Also, some of the post-hoc explanation methods that we have studied, are model specific, or only have faster run-time on a specific group of ML models. For example despite the simplicity and faster running-time TreeInterpreter, it is limited to some traditional tree-based ML models (mainly Decision Tree and Random Forest). In contrast LIME, which is significantly slower in generating local explanations, has the advantage of being model agnostic. Similarly, Shapley explanations have several variations, including a model agnostic algorithm (SHAP) and another, TreeSHAP, that is specialized for tree based models.

REFERENCES

[1] Ugo Cupcic, "Grasping dataset, shadow robot company," *URL:https://www.kaggle.com/ugocupcic/grasping-dataset*, 2019.

[2] Ugo Cupcic, "How i taught my robot to realize how bad it was at holding things, shadow robot company," *URL:https://www.wevolver.com/article/*, 2019.

[3] Yehuda Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2008, pp. 426–434.

[4] Behnoush Abdollahi and Olfa Nasraoui, "Explainable matrix factorization for collaborative filtering," in *Proceedings of the 25th International Conference Companion on World Wide Web*, 2016, pp. 5–6.

[5] Behnoush Abdollahi and Olfa Nasraoui, "Using explainability for constrained matrix factorization," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, 2017, pp. 79–83.

[6] Khalil Damak, Sami Khenissi, and Olfa Nasraoui, "Debiased explainable pairwise ranking from implicit feedback," in *Fifteenth ACM Conference on Recommender Systems*, 2021, pp. 321–331.

[7] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *arXiv preprint arXiv:1703.09312*, 2017.

[8] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin, "Unbiased offline recommender evaluation for missing-not-at-random implicit feedback," in *Proceedings of the 12th ACM Conference on Recommender Systems*, 2018, pp. 279–287.

[9] Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker, "Accurate intelligible models with pairwise interactions," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013, pp. 623–631.

[10] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad, "Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission," in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, pp. 1721–1730.

[11] Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana, "Interpretml: A unified framework for machine learning interpretability," *arXiv preprint arXiv:1909.09223*, 2019.

[12] Paul Resnick and Hal R Varian, "Recommender systems," *Communications of the ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[13] Robin Burke, "Hybrid web recommender systems," in *The adaptive web*, pp. 377–408. Springer, 2007.

[14] Francesco Ricci, Lior Rokach, and Bracha Shapira, "Introduction to recommender systems handbook," in *Recommender systems handbook*, pp. 1–35. Springer, 2011.

[15] David M Pennock, Eric J Horvitz, Steve Lawrence, and C Lee Giles, "Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach," *arXiv preprint arXiv:1301.3885*, 2013.

[16] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He, "Bias and debias in recommender system: A survey and future directions," *arXiv preprint arXiv:2010.03240*, 2020.

[17] Himan Abdollahpouri and Masoud Mansoury, "Multi-sided exposure bias in recommendation," *arXiv preprint arXiv:2006.15772*, 2020.

[18] Malcolm Gladwell, *Outliers: The story of success*, Little, Brown, 2008.

[19] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan, "A survey on bias and fairness in machine learning," *arXiv preprint arXiv:1908.09635*, 2019.

[20] Batya Friedman and Helen Nissenbaum, "Bias in computer systems," *ACM Transactions on Information Systems (TOIS)*, vol. 14, no. 3, pp. 330–347, 1996.

[21] Anja Lambrecht and Catherine Tucker, "Algorithmic bias? an empirical study of apparent gender-based discrimination in the display of stem career ads," *Management Science*, vol. 65, no. 7, pp. 2966–2981, 2019.

[22] Amit Datta, Michael Carl Tschantz, and Anupam Datta, "Automated experiments on ad privacy settings: A tale of opacity, choice, and discrimination," *Proceedings on privacy enhancing technologies*, vol. 2015, no. 1, pp. 92–112, 2015.

[23] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher, "Controlling popularity bias in learning-to-rank recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, 2017, pp. 42–46.

[24] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma, "Correcting popularity bias by enhancing recommendation neutrality.," in *RecSys Posters*, 2014.

[25] Adit Krishnan, Ashish Sharma, Aravind Sankar, and Hari Sundaram, "An adversarial approach to improve long-tail performance in neural collaborative filtering," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1491–1494.

[26] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Yong Li, and Depeng Jin, "Disentangling user interest and popularity bias for recommendation with causal embedding," *arXiv preprint arXiv:2006.11011*, 2020.

[27] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims, "Recommendations as treatments: Debiasing learning and evaluation," *arXiv preprint arXiv:1602.05352*, 2016.

[28] Saúl Vargas and Pablo Castells, "Rank and relevance in novelty and diversity metrics for recommender systems," in *Proceedings of the fifth ACM conference on Recommender systems*, 2011, pp. 109–116.

[29] Chang Nho Cho, Ji Tae Hong, and Hong Ju Kim, "Neural network based adaptive actuator fault detection algorithm for robot manipulators," *Journal of Intelligent & Robotic Systems*, vol. 95, no. 1, pp. 137–147, 2019.

[30] Jin-Ho Shin and Ju-Jang Lee, "Fault detection and robust fault recovery control for robot manipulators with actuator failures," in *Proceedings 1999 IEEE International Conference on Robotics and Automation*. IEEE, 1999, vol. 2, pp. 861–866.

[31] Shingo Kitagawa, Kentaro Wada, Shun Hasegawa, Kei Okada, and Masayuki Inaba, "Multi-stage learning of selective dual-arm grasping based on obtaining and pruning grasping points through the robot experience in the real world," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7123–7130.

[32] Cosimo Della Santina, Visar Arapi, Giuseppe Averta, Francesca Damiani, Gaia Fiore, Alessandro Settimi, Manuel G Catalano, Davide Bacciu, Antonio Bicchi, and Matteo Bianchi, "Learning from humans how to grasp: a data-driven architecture for autonomous grasping with anthropomorphic soft hands," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1533–1540, 2019.

[33] Riccardo Muradore and Paolo Fiorini, "A pls-based statistical approach for fault detection and isolation of robotic manipulators," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 8, pp. 3167–3175, 2011.

[34] Carlo Cecati, "A survey of fault diagnosis and fault-tolerant techniques—part ii: Fault diagnosis with knowledge-based and hybrid/active approaches," 2015.

[35] Mark A Kramer and BL Palowitch Jr, "A rule-based approach to fault diagnosis using the signed directed graph," *AIChE journal*, vol. 33, no. 7, pp. 1067–1078, 1987.

[36] Wen-Shing Lee, Doris L Grosh, Frank A Tillman, and Chang H Lie, "Fault tree analysis, methods, and applications a review," *IEEE transactions on reliability*, vol. 34, no. 3, pp. 194–203, 1985.

[37] V Esposito Vinzi, Wynne W Chin, Jörg Henseler, Huiwen Wang, et al., *Handbook of partial least squares*, vol. 201, Springer, 2010.

[38] DF Frey and RA Pimentel, "Principal component analysis and factor analysis," 1978.

[39] Vladimir N Vapnik, "The nature of statistical learning," *Theory*, 1995.

[40] Lotfi A Zadeh, "Fuzzy logic," *Computer*, vol. 21, no. 4, pp. 83–93, 1988.

[41] Peter C Cheeseman, Matthew Self, James Kelly, Will Taylor, Don Freeman, and John C Stutz, "Bayesian classification.," in *AAAI*, 1988, vol. 88, pp. 607–611.

[42] Thomas Cover and Peter Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[43] Stuart Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[44] James MacQueen et al., "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Oakland, CA, USA, 1967, vol. 1, pp. 281–297.

[45] Xuewu Dai and Zhiwei Gao, "From model, signal to knowledge: A data-driven perspective of fault detection and diagnosis," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 4, pp. 2226–2238, 2013.

[46] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[47] Filipe Veiga, Jan Peters, and Tucker Hermans, "Grip stabilization of novel objects using slip prediction," *IEEE transactions on haptics*, vol. 11, no. 4, pp. 531–542, 2018.

[48] Rui Li and Hong Qiao, "A survey of methods and strategies for high-precision robotic grasping and assembly tasks—some new trends," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 6, pp. 2718–2732, 2019.

[49] Visar Arapi, Yujie Zhang, Giuseppe Averta, Manuel G Catalano, Daniela Rus, Cosimo Della Santina, and Matteo Bianchi, "To grasp or not to grasp: an end-to-end deep-learning approach for predicting grasping failures in soft hands," in *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*. IEEE, 2020, pp. 653–660.

[50] Donghyun Park, Seulgi Kim, Yelin An, and Jae-Yoon Jung, "Lired: A light-weight real-time fault detection system for edge computing using lstm recurrent neural networks," *Sensors*, vol. 18, no. 7, pp. 2110, 2018.

[51] Leo Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[52] Meinard Müller, "Dynamic time warping," *Information retrieval for music and motion*, pp. 69–84, 2007.

[53] Silvia Sapora, *Grasp Quality Deep Neural Networks for Robotic Object Grasping*, Ph.D. thesis, Imperial College London, 2019.

[54] Christoph Molnar, "A guide for making black box models explainable," *URL: https://christophm. github. io/interpretable-ml-book*, 2018.

[55] Trevor Hastie, Robert Tibshirani, and Jerome Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Springer Science & Business Media, 2009.

[56] Jan Salomon Cramer, "The origins of logistic regression," 2002.

[57] Kyle Chung, "On model explainability, from lime, shap, to explainable boosting," *URL:https://everdark.github.io/k9/notebooks/ml/model-explain/model-explain.nb.html*, 2019.

[58] Lior Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 1–39, 2010.

[59] Zachary C Lipton, "The mythos of model interpretability," *Queue*, vol. 16, no. 3, pp. 31–57, 2018.

[60] Mark Craven and Jude W Shavlik, "Extracting tree-structured representations of trained networks," in *Advances in neural information processing systems*, 1996, pp. 24–30.

[61] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert MÃžller, "How to explain individual classification decisions," *Journal of Machine Learning Research*, vol. 11, no. Jun, pp. 1803–1831, 2010.

[62] Igor Kononenko et al., "An efficient explanation of individual classifications using game theory," *Journal of Machine Learning Research*, vol. 11, no. Jan, pp. 1–18, 2010.

[63] Marco Tulio Ribeiro, Sameer Singh, and Carlos 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*, 2016, pp. 1135–1144.

[64] Aaron Fisher, Cynthia Rudin, and Francesca Dominici, "All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously," *Journal of Machine Learning Research*, vol. 20, no. 177, pp. 1–81, 2019.

[65] Scott M Lundberg and Su-In Lee, "A unified approach to interpreting model predictions," in *Advances in neural information processing systems*, 2017, pp. 4765–4774.

[66] Ando Saabas, "Interpreting random forests," *Diving into data*, vol. 24, 2014.

[67] Ando Saabas, "Treeinterpreter library (2019)," *https://github.com/andosa/treeinterpreter*.

[68] Trevor Hastie and Robert Tibshirani, "Generalized additive models: some applications," *Journal of the American Statistical Association*, vol. 82, no. 398, pp. 371–386, 1987.

[69] Yin Lou, Rich Caruana, and Johannes Gehrke, "Intelligible models for classification and regression," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 150–158.

[70] Dan Frankowski, "A gentle introduction to ga2ms, a white box model," *URL:https://blog.fiddler.ai/2019/06/a-gentle-introduction-to-ga2ms-a-white-box-model/*, 2019.

[71] Scott M Lundberg, Gabriel G Erion, and Su-In Lee, "Consistent individualized feature attribution for tree ensembles," *arXiv preprint arXiv:1802.03888*, 2018.

[72] Katsushige Fujimoto, Ivan Kojadinovic, and Jean-Luc Marichal, "Axiomatic characterizations of probabilistic and cardinal-probabilistic interaction indices," *Games and Economic Behavior*, vol. 55, no. 1, pp. 72–99, 2006.

[73] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in neural information processing systems*, 2017, pp. 3146–3154.

[74] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[75] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen, *Classification and regression trees*, CRC press, 1984.

[76] Leo Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[77] Stacey Ronaghan, "The mathematics of decision trees, random forest and feature importance in scikit-learn and spark," *Online: https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3*, 2018.

[78] Maurice G Kendall, "The treatment of ties in ranking problems," *Biometrika*, vol. 33, no. 3, pp. 239–251, 1945.

[79] William Webber, Alistair Moffat, and Justin Zobel, "A similarity measure for indefinite rankings," *ACM Transactions on Information Systems (TOIS)*, vol. 28, no. 4, pp. 1–38, 2010.

[80] Jean Dickinson Gibbons and Subhabrata Chakraborti, "Nonparametric statistical inference fourth edition, revised and expanded," *STATISTICS TEXTBOOKS AND MONOGRAPHS*, vol. 168, 2003.

[81] Dawen Liang, Laurent Charlin, James McInerney, and David M Blei, "Modeling user exposure in recommendation," in *Proceedings of the 25th international conference on World Wide Web*, 2016, pp. 951–961.

[82] Wenlong Sun, Sami Khenissi, Olfa Nasraoui, and Patrick Shafto, "Debiasing the human-recommender system feedback loop in collaborative filtering," in *Companion Proceedings of The 2019 World Wide Web Conference*, 2019, pp. 645–651.

[83] F Maxwell Harper and Joseph A Konstan, "The movielens datasets: History and context," *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.

[84] Sebastiano Vigna, "A weighted correlation index for rankings with ties," in *Proceedings of the 24th international conference on World Wide Web*, 2015, pp. 1166–1176.

[85] A. Alvanpour, S. K. Das, C. K. Robinson, O. Nasraoui, and D. Popa, "Robot failure mode prediction with explainable machine learning," in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, 2020, pp. 61–66.

CURRICULUM VITAE

ANESEH ALVANPOUR
aalvanpour@gmail.com

PROFESSIONAL SUMMARY

Skilled in Data Science, Algorithms, and Statistics, with
Strong Research Background in Fairness and Interpretability
in Machine Learning, specifically in Recommender Systems.
Experienced in Building Predictive Models on Healthcare Data
and explored Data Augmentation Methods for Tabular Data including
Generative Adversarial Networks (GANs) in Finance.

SKILLS

- **Technical skills**: Data Science, Machine Learning, Neural Networks and Deep Learning, Data Visualization, Statistics
- **Programming skills**: Python, Scikit-learn, Pandas, R, MySQL
- **Fairness and Explainability toolkits and approaches:**
  - Fairlearn by Microsoft, AI Fairness 360 by IBM, Aequitas
  - InterpretML by Microsoft, AI Explainability 360 by IBM, SHAP, LIME, TreeInterpreter
- **Business Intelligence**: Tableau, SAS Visual Analytics
- **Management skills:** Teamwork, Teaching, Decision-making
- **Transferable skills:** Communication, Analytical, Problem-solving, Creative thinking
- **Certifications:**
  - **Neural Networks and Deep Learning,** deeplearning.ai on Coursera August 2018
  - **Data Science Certificate,** University of Louisville May 2018
- **Languages:** Persian(professional), English(professional)

EDUCATION

Ph.D. Computer Science May 2022
University of Louisville; Louisville, KY

M.Sc. Computer Science May 2017
University of Louisville; Louisville, KY

B.S. Information and Communication Technology Engineering (ICT) September 2010
Kashani University; Qazvin, Iran

WORK HISTORY & RESEARCH

WORK

Machine Learning Modeling March 2022- present
Discover Financial Services

Graduate Research Assistant August 2019 – March 2022

Advanced Automation and Robotics Research Institute (LARRI), University of Louisville, Louisville, KY

- Robot Failure Mode Prediction with Explainable Machine Learning.
  - Study different Explainable Machine Learning models to have a better understanding of algorithmic decision systems in robotics and to be able to design an intelligent variable autonomy based architecture with more trustable interactions between human and the robot.
  - Ranking similarity analysis of Post-hoc Explanation methods including SHAP, LIME and Treeinterpreter.

Senior Data Scientist Intern June 2020 - August 2020

Discover Financial Services

- Research on Data Augmentation Methods for Tabular Data.
  - Explored oversampling techniques to generate synthetic tabular data, including Generative Adversarial Networks (GANs) and SMOTE variants, to provide solutions for the highly imbalanced class problem.

Data Scientist Intern May 2019 - August 2019

The Rawlings Group, La Grange, KY

- Applied Machine Learning methods to provide solutions for the business to reduce the cost of healthcare.
  - Enhancing money recovery from health insurance claims by building predictive models and extracting important features and saved time in the insurance claims recovery process.

Graduate Research Assistant September 2017 - May 2019

Graduate School, University of Louisville, Louisville, KY

- Data Analysis and Visualization
  - Applied data preprocessing techniques and statistical tests to educational and financial data regarding graduate students.
  - Produce informative dashboards and reports.

Graduate Student Assistant August 2016 - September 2017

REACH (Resources for Academic Achievement) Computer Resource Center, University of Louisville, Louisville, KY

- Assisted in the management, hiring and training of tutors for Computer Science courses.
  - Scheduled 15 tutors' working hours and monitored their work.
  - Evaluated candidates for tutoring positions in the interviews.
  - Conducted training sessions for new tutors through workshops and practical tests.
  - Produced information sessions to introduce REACH to more than 200 hundred undergraduate students in the College of Business and Department of Computer Science.
- Assisted undergraduate students with general computer-related issues, as well as specific programming questions in Computer Science courses.

Graduate Teaching Assistant September 2014 - May 2016

Computer Science and Engineering, University of Louisville, Louisville, KY

- Assisted in the teaching of the C programming course to 60 undergraduate students.
  - Guided students to solve their problems in C programming and graded homework and exams.

RESEARCH

- Studying Accuracy, Transparency, and Fairness trade-offs in recommendation systems.
  - Exploring the effect of popularity debiasing on Matrix Factorization's performance and providing more novel and diverse recommendations.
- Investigate the trade-off between Accuracy and Interpretability of Machine Learning methods in robotic grasp failure prediction.
  - Adding explanations can bridge the communication gap between human and the robot.
  - The explanations agree with concepts from the mechanical design of graspers.
  - The human operator can use a predicted failure's explanation to intervene and debug the system.
- Explored correlation between alcohol usage and socio-demographics attributes among students by Clustering, Association Rule Mining and Linear Regression Models.

- Predicted the hospital readmission for diabetic patients by Decision Tree Classifier and Random Forest.
- Applied interpretability of the Decision Tree Classifier to explain why a certain exceptional machine-learned decision was made incorrectly.

PAPERS

- A. Alvanpour, S. K. Das, C. K. Robinson, O. Nasraoui, and D. Popa, "Robot failure mode prediction with explainable machine learning," in 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), pp. 61–66, 2020.

COURSEWORK

Algorithms, Data Mining, Neural Networks and Deep Learning (Course 1 on Coursera), Database, Statistical Data Analysis, Web Mining for E-Commerce and Information Retrieval

HONOR AND AWARDS

Graduate Dean's Citation Award

May 2022

- Award for top 10% of graduate students in each program, Graduate School,
- University of Louisville, KY

Grace Hopper Celebration Student Scholarship July 2021, September 2019, May 2018
- Award to attend the world's largest gathering of women technologists by AnitaB.org and ACM

Computer Science and Engineering Arthur M. Riehl Award April 2021
- Award to a graduate student with excellent academic performance and contributions to the department activities, Speed School of Engineering, University of Louisville, KY

CRA-WP Graduate Cohort Workshops for Women Scholarship April 2021
- Award to attend the Graduate Cohort Workshops for Women by the Computing Research Association

Speed School Research Exposition Award April 2019
- 2nd place poster winner in PhD category, Speed School of Engineering, University of Louisville, KY

Graduate Dean's Citation Award March 2017
- Award for top 10% of graduate students in each program, Graduate School, University of Louisville, KY

VOLUNTEER WORK/COMMUNITY SERVICE

Program Committee at the FAccTRec Workshop on Responsible Recommendation (RecSys 2021) September 2021
15th ACM Conference on Recommender Systems

Volunteer at the ACM FAccT Conference March 2021
4th ACM Conference on Fairness, Accountability, and Transparency

Mentoring 4 undergraduate students on Bias and Explainability Research January 2021-present
Computer Science and Engineering, University of Louisville, KY
- Helping them to start their research, conducting experiments and presenting their results to the team

Super-volunteer at Women in Machine Learning Workshop (WiML 2020) at the NeurIPS Conference December 2020
5th Annual Conference on Neural Information Processing Systems

Chair of Machine Learning and Data Mining Session at IEEE CASE Conference August 2020
IEEE 16th International Conference on Automation Science and Engineering

Reviewer at IEEE CASE Conference March 2020

IEEE 16th International Conference on Automation Science and Engineering

NSF Research Experiences for Teachers (RET) Site in Big Data and Data Science June 2018

C***omputer Science and Engineering, University of Louisville, KY***

- Collaborated in the training of 10 high school teachers in Big Data and Machine Learning concepts and tools

Knowledge Discovery and Web Mining Lab's Leader in Engineering Exposition March 2018

Speed School of Engineering, University of Louisville, KY

- Gave a laboratory tour for over 80 people especially K-12 students and parents to encourage them to the STEM fields.

Vice President of Iranian Students Organization at University of Louisville October 2015 - August 2017

- Helped to plan and hold two Iranian cultural events with more than 400 hundred guests and assisted with new Iranian students' orientation.