5-2024

# In-training explainability frameworks to make black-box machine learning models more explainable.

Asuman Cagla Acun
*University of Louisville*

# IN-TRAINING EXPLAINABILITY FRAMEWORKS TO MAKE BLACK-BOX MACHINE LEARNING MODELS MORE EXPLAINABLE

By

Asuman Cagla Acun
B.S. in Computer Engineering, Hacettepe University, Turkey, 2013
M.Sc. in Computer Science, University of Louisville, USA, 2017

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 Science and Engineering
University of Louisville
Louisville, Kentucky

May 2024

IN-TRAINING EXPLAINABILITY FRAMEWORKS TO MAKE BLACK-BOX MACHINE LEARNING MODELS MORE EXPLAINABLE

By

Asuman Cagla Acun
B.S. in Computer Engineering, Hacettepe University, Turkey, 2013
M.Sc. in Computer Science, University of Louisville, USA, 2017

A Dissertation Approved On

April 24, 2024

by the following Dissertation Committee

---

Dr. Olfa Nasraoui, Dissertation Director

---

Dr. Hichem Frigui

---

Dr. Dan Popa

---

Dr. Nihat Altiparmak

---

Dr. Sabur Baidya

# ACKNOWLEDGEMENTS

I am deeply grateful for the persistent support and belief of my advisor, Dr. Olfa Nasraoui, whose guidance and encouragement have been invaluable to my journey. She always believed in me and did not let me lose my faith in my most difficult moments.

I greatly appreciate my supportive family, my mother and father, and my beloved sisters Bilge and Isin. They believed in me and supported me through the toughest times. I could not have finished this journey without their unwavering love and encouragement.

I also want to thank my committee members, Dr. Dan Popa, Dr. Nihat Altiparmak, Dr. Hichem Frigui, and Dr. Sabur Baidya, for taking the time to review my work and for taking the time to review my dissertation and provide valuable feedback.

I want to thank my dear son Everest, who was in my womb during my first classes, for accompanying me and for showing understanding during the periods when I was working intensely. Throughout this journey, we have always been together in all the difficulties we experienced, and we will continue to be together. I love you so much!

Finally, I would like to acknowledge the Republic of Turkey for supporting my research through the Study Abroad Program of the Ministry of National Education, and the National Science Foundation for supporting my research through Grant 2026584 FW-HTF-RM: Enhancing Future Work of Nursing Professionals through Collaborative Human-Robot Interfaces while I was working as a research assistant at Louisville Automation & Robotics Research Institute (LARRI).

# ABSTRACT

## IN-TRAINING EXPLAINABILITY FRAMEWORKS TO MAKE BLACK-BOX MACHINE LEARNING MODELS MORE EXPLAINABLE

Asuman Cagla Acun

April 24, 2024

Despite ongoing efforts to make black-box machine learning models more explainable, transparent, and trustworthy, there is growing advocacy for using only inherently interpretable models for high-stake decision-making. Post-hoc explanations have recently been criticized for learning surrogate models that may not accurately reflect the actual mechanisms of the original model and for adding computational burden at prediction time. We propose two novel explainability approaches to address these limitations: pre-hoc explainability and co-hoc explainability. These approaches integrate explanations derived from an inherently interpretable white-box model into the learning stage of the black-box model without compromising accuracy. Unlike post-hoc methods, our approach does not rely on random input perturbation or only post-hoc training. We extend our pre-hoc and co-hoc frameworks to generate instance-specific explanations by incorporating the Jensen-Shannon divergence as a regularization term while capturing the local behavior of the black-box model. This extension allows our methods to provide local explanations that are faithful to the model's behavior and consistent with the explanations generated by the global explainer model. We introduce a two-phase approach, where the first phase focuses on training the models for fidelity, and the second phase generates local explanations by fine-tuning the explainer model within the neighborhood of the instance being explained.

Experiments on three benchmark datasets from different domains (credit risk scoring, movie recommendations, and robotic grasper failure detection) demonstrate the advantages of our techniques in terms of global and local fidelity without compromising accuracy. Our methods avoid the pitfalls of surrogate modeling, making them more scalable, robust, and reliable compared to post-hoc techniques like LIME. Moreover, our co-hoc learning framework enhances the accuracy of white-box models, which are learned to explain the black-box predictor. The white-box model achieves significantly higher prediction accuracy after the co-hoc learning process, highlighting the potential of the co-hoc in-training approach to improve the performance of white-box models, which are essential and required in specific high-risk and regulated application tasks in healthcare and legal decision-making.

Our approaches provide more faithful and consistent explanations at a lower computational cost than LIME. Our theoretically derived methods are further shown to balance accuracy and interpretability through empirically regularized learning. The proposed frameworks offer a promising direction for making machine learning models more transparent and trustworthy while maintaining high prediction accuracy.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii

ABSTRACT iv
LIST OF TABLES ix
LIST OF FIGURES xi
LIST OF ALGORITHMS xiv

CHAPTERS

1 INTRODUCTION . . . . . 1
1.1 Problem Statement . . . . . 3
1.2 Research Contributions . . . . . 4
1.3 Document Organization . . . . . 6

2 LITERATURE REVIEW . . . . . 8
2.1 Inherently Interpretable Models . . . . . 8
2.1.1 Linear Models . . . . . 8
2.1.2 Decision Trees . . . . . 10
2.2 Post-hoc Explainability . . . . . 11
2.2.1 LIME (Local Interpretable Model-Agnostic Explanations) . . . . 11
2.2.2 SHAP (SHapley Additive exPlanations) . . . . . 13
2.3 Model-Specific Explainability (Explainability by Design) . . . . . 14
2.4 Existing In-Training Explainability Techniques . . . . . 16
2.5 Explanation Types . . . . . 17
2.5.1 Rule-based Explanations . . . . . 18
2.5.2 Feature-based Explanations . . . . . 18
2.5.3 Concept-based Explanations . . . . . 20
2.5.4 Instance-based Explanations . . . . . 21
2.5.5 Local Explanations . . . . . 22

2.5.6 Global Explanations . . . 23
2.5.7 Multi-Model Explanations . . . 24
2.6 Explanation Evaluation Metrics . . . 25
2.7 Factorization Machine Model . . . 28
2.8 Transparency in Factorization Machine Model . . . 31
2.9 Summary . . . 32

3 A NEW EXPLAINABLE MACHINE LEARNING FRAMEWORK BASED ON PRE-HOC AND CO-HOC EXPLAINABILITY . . . 33
3.1 Introduction . . . 33
3.2 Problem Statement . . . 33
3.3 Enforcing Fidelity . . . 34
3.4 Pre-hoc Explainability Framework . . . 37
3.5 Co-hoc Explainability Framework . . . 39
3.6 Generating Explanations . . . 40
3.7 Experimental Results . . . 42
3.7.1 Research Questions . . . 42
3.7.2 Datasets . . . 42
3.7.3 Evaluation Protocols . . . 43
3.7.4 Baselines . . . 43
3.7.5 Parameter Settings . . . 43
3.7.6 Results and Discussion . . . 44
3.8 Summary . . . 49

4 OPTIMIZING IN-TRAINING EXPLAINABLE MACHINE LEARNING FOR LOCAL EXPLAINABILITY . . . 50
4.1 Introduction . . . 50
4.2 Problem Statement . . . 50
4.3 Optimizing Local Explainability Using the Neighbors of an Instance . . . 54
4.3.1 PHASE 1: Integrating Local Explainability with Neighbors in Training . . . 56
4.3.2 PHASE 2: Generating Local Explanations . . . 59
4.4 Comparison to Classical Post-hoc Explainability Methods such as LIME 61

4.5 Experimental Results . . . . . 62
4.5.1 Research Questions . . . . . 62
4.5.2 Datasets . . . . . 62
4.5.3 Evaluation Protocols . . . . . 63
4.5.4 Baselines . . . . . 65
4.5.5 Parameter Settings . . . . . 65
4.5.6 Results and Discussion . . . . . 65
4.5.7 Examples of Explanations . . . . . 81
4.5.8 Summary . . . . . 86

5 CONCLUSION . . . . . 88
5.1 Summary of Contributions . . . . . 88
5.2 Limitations and Future Work . . . . . 90
5.3 Conclusion . . . . . 92

REFERENCES 93

CURRICULUM VITAE 102

# LIST OF TABLES

TABLE Page

3.1 Proposed Explainability Frameworks, $\lambda_1$ comparison in prediction accuracy (AUC) and explainability (Fidelity) on the three datasets that were described in experimental settings. Higher AUC and Fidelity is better. . . . . 44

3.2 Model comparison in terms of prediction accuracy and fidelity of explainability on the three real-world datasets, two interaction datasets, ML100k and ML1M, and one tabular, HELOC dataset. All evaluation metrics are computed with, respectively, for datasets $\lambda_1 = 0.75, 0.5, 1$ and batch size $n = 64, 2056, 64$. The explainer is the white-box model, BB is the predictor black-box model. The best results are in **bold**. Higher AUC and Fidelity is better. . . . . 45

4.1 Notation . . . . . 51

4.2 HELOC Dataset results. Comparison with LIME based on neighborhood fidelity and stability results. $\lambda = 0.25$, $k = 10$. . . . . 73

4.3 Grasping Dataset results. Comparison with LIME based on neighborhood fidelity and stability results. $\lambda = 0.25$, $k = 10$. One hundred test instances in the neighborhood, an average of 5 runs. . . . . 74

4.4 ML-100k Dataset results. Comparison of the Pre-hoc Framework with $k = 10$, for $\lambda = \{0.01, 0.05, 0.1, 0.25, 0.5, 1\}$ in point fidelity, neighborhood fidelity, and stability results. “Reg" means that regularization was used. . . . . 75

4.5 HELOC Dataset results on Pre-hoc for varying neighborhood size based on neighborhood fidelity, stability, and computation cost results. $\lambda = 0.25$. . . . 77

4.6 HELOC Dataset. Computation time comparison for generating explanations on a test dataset of 100 instances. The average time is the computation time to explain a single instance. . . . . 77

4.7 Grasping Dataset. Computation time comparison for generating explanations on a test dataset of 100 instances. The average (Avg) time is the computation time to explain a single instance. Results are based on five runs. . . . . 79
4.8 Computation Cost comparison using CPU and GPU in Pre-hoc Explainability Framework. . . . . 80

# LIST OF FIGURES

FIGURE Page

1.1 XAI Concept [1] . . . . . 2

2.1 The image illustrates a white-box model, represented by a semi-transparent box with visible gears inside on the left, and a black-box model, depicted as a label-free opaque black box on the right. Source: generated using DALL-E. 9

3.1 Proposed Explainability Frameworks . . . . . 34

3.2 Training Phase of Pre-hoc Explainability Framework . . . . . 37

3.3 Training Phase of Co-hoc Explainability Framework . . . . . 38

3.4 Pre-hoc Explainability Framework Comparison in Accuracy AUC for different $\lambda_1$ on the ml-100k (a), ml-1M (b), HELOC (c) datasets. Pre-hoc Predictor is our proposed model, BB is the original black-box predictor model, WB is the explainer model. . . . . 46

3.5 Co-hoc Explainability Framework Comparison in Accuracy AUC for different $\lambda_1$ on the ml-100k (a), ml-1M (b), HELOC (c) datasets. Co-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model. . . . . 46

4.1 Pre-hoc Local Explainable Machine Learning Framework consists of two phases. Phase 1: Training for Fidelity is the training phase that optimizes the agreement between the white-box explainer and black-box predictor models, quantified by fidelity. Phase 2: Generating a local explanation for a new test instance by fine-tuning the white-box explainer model within its neighboring training instances. . . . . 52

4.2 Co-hoc Local Explainable Machine Learning Framework consists of 2 phases. Phase 1: Training for Fidelity is the training phase of the framework that co-optimizes the agreement between the white-box explainer and black-box predictor models, quantified by fidelity. Phase 2: Generating a local explanation for a new test instance by fine-tuning the white-box explainer model within its neighboring training instances. . . . . 53
4.3 HELOC Dataset. Pre-hoc Explainability Framework Comparison in Accuracy AUC and Fidelity AUC for different explainability regularization $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Pre-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model. . . . . 68
4.4 HELOC Dataset. Co-hoc Explainability Framework (a) Accuracy AUC (b) Fidelity AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Co-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model. . . . . 68
4.5 Grasping Dataset. Pre-hoc Local Explainability Framework (a) Accuracy Fidelity (b) Fidelity AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Pre-hoc Local Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model. . . . . 69
4.6 Grasping Dataset. Co-hoc Local Explainability Framework (a) Accuracy Fidelity (b) Fidelity AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Co-hoc Local Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model. . . . . 70
4.7 ML-100k Dataset. Pre-hoc Explainability Framework Comparison in Accuracy AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Pre-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model. . . . . 71
4.8 ML-100k Dataset. Co-hoc Explainability Framework Comparison in Accuracy AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Co-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model. . . . . 72

4.9 Grasping and HELOC Datasets results. Comparison with LIME based on neighborhood fidelity and stability results. $\lambda = 0.25$, $k = 10$. One hundred test instances in the neighborhood, an average of 5 runs. . . . . 73
4.10 ML-100k Dataset results. Comparison of the Pre-hoc Framework with $k = 10$, for $\lambda = \{0.01, 0.05, 0.1, 0.25, 0.5, 1\}$ in point fidelity, neighborhood fidelity, and stability results. "Reg" means that regularization was used. . . . . 75
4.11 HELOC Dataset results on Pre-hoc for varying neighborhood size based on neighborhood fidelity, stability, and computation cost results. $\lambda = 0.25$. . . . . 76
4.12 Grasping Data Set Explanation Example: Top 10 Feature Importance scores from the global explanation of the Pre-hoc framework, showing that increased effort exerted in Joint 2 of Finger 3 contributes tremendously to producing grasp failure, which is in contrast to increased efforts exerted at joint 2 of Fingers 1 and 2 and joint 1 of Finger 3, that lead to reducing grasp failure. . 82
4.13 Grasping Dataset Local Explanation Example for Test Instance 143. The true label is predicted correctly by the predictor model in the Pre-hoc Framework, and the plotted Top10 feature importance scores are generated by the explainer model in the Pre-hoc Framework. . . . . 83
4.14 HELOC Dataset Global Explanation Example . . . . . 84
4.15 HELOC Dataset Local Explanation Example for Test Instance 1. The true label is predicted correctly by the predictor model in the Pre-hoc Framework, and the plotted Top10 feature importance scores are generated by the explainer model in the Pre-hoc Framework. . . . . 84
4.16 HELOC Dataset Local Explanation Example for Test Instance 50. The true label is **not predicted correctly** by predictor model in Pre-hoc Framework and the plotted Top10 feature importance scores generated by explainer model in Pre-hoc Framework. . . . . 85

# LIST OF ALGORITHMS

ALGORITHM Page

2.1 Sparse Linear Explanations using LIME . . . . . 12

3.1 Pre-hoc Explainability Framework . . . . . 38

3.2 Co-hoc Explainability Framework . . . . . 40

3.3 Feature Importance Score Calculation . . . . . 41

4.1 Training PHASE 1 for Pre-hoc: Integrating Local Explainability with Neighbors in Training . . . . . 57

4.2 Training PHASE 1 for Co-hoc: Integrating Local Explainability with Neighbors in Training . . . . . 58

4.3 Testing PHASE 2: Computing Local Explanations . . . . . 59

# CHAPTER 1

# INTRODUCTION

Machine learning models are increasingly used to support decision-making in various fields, from personalized medical diagnosis to credit risk assessment and criminal justice. However, the increasing reliance on powerful black-box models raises concerns about their transparency, interpretability, and trustworthiness [2–4]. Understanding why a model made a particular prediction is crucial to supporting auditing models, detecting potential biases and errors, and, in turn, supporting model accountability and fairness.

Explainable Artificial Intelligence (XAI) has emerged as a new research area that focuses on machine learning interpretability. The goal is to build interpretable models that will generate high-performing machine learning models [5] and thus enable human users to understand the models and trust them.

In machine learning, the term explainability still lacks a common meaning, and the capability varies from application to application. Interpretability is often used instead. However, traditionally, explainability or interpretability refers to the ability of an artificial intelligence system to be understood by humans [6].

Explainable AI helps build trust in machine learning systems by providing insights into how models make decisions. This is particularly important in high-stakes domains such as healthcare, finance, and criminal justice, where the consequences of incorrect or biased decisions can be severe. When users understand how a model arrives at a particular output, they are more likely to trust and rely on the system [7, 8]. Explanations can help identify errors, biases, and unexpected behaviors in machine learning models. Developers can debug and improve their models by understanding how features influence predictions, leading to more accurate and reliable systems. [9, 10]. In some domains, there are legal and regulatory requirements to explain algorithmic decisions. For example, the European Union's General Data Protection Regulation (GDPR) includes a "right to explanation" for individuals subject to automated decision-making. Explainable AI techniques can help organizations comply with these regulations [11, 12]. Explainable AI can also help uncover biases and unfairness in

machine learning models [13, 14]. Furthermore, explainable AI enables effective human-AI collaboration by providing a common understanding between humans and machines [15, 16].

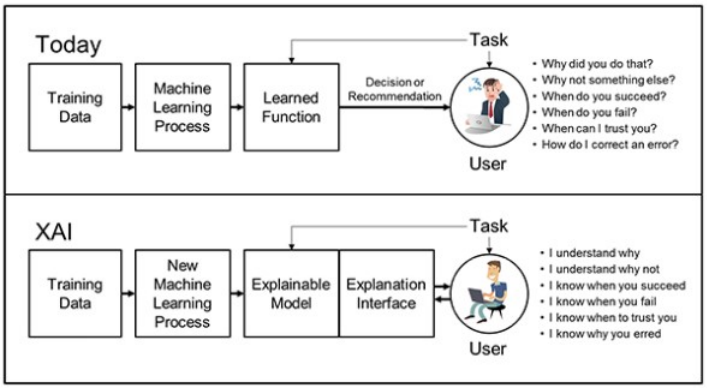



Figure 1.1: XAI Concept [1]

Several approaches have been proposed to explain black-box models, ranging from local methods that provide explanations for individual predictions to global methods that aim to capture the model's overall behavior. Post-hoc explanations, such as LIME (Local Interpretable Model-Agnostic Explanations) [17], SHAP (Shapley Additive Explanations) [18], and Grad-CAM (Gradient Weighted Class Activation Mapping) [19], have gained popularity in recent years as a way to explain black-box models by perturbing the input data and learning a surrogate model that approximates the original model's behavior locally. Although these methods can effectively generate explanations, they have been criticized for several reasons. First, the explanations may not reflect the true mechanisms of the original model but rather a simplified version that is easier to interpret [20]. Second, the surrogate model may not be faithful to the behavior of the original model in some cases, leading to potentially misleading explanations and being open to adversarial attacks [21]. Third, the input data's perturbation can alter the features' semantics, rendering the explanations invalid or misleading and unstable explanations that arise with models already trained [22] [23].

To address these limitations, some researchers have proposed using inherently interpretable models, such as decision trees or linear models, instead of black-box models for high-stakes decision-making [24]. However, this approach may come at the cost of reduced prediction accuracy, as interpretable models may not be able to capture the complexity of some datasets and black-box models. Moreover, this approach cannot be applied to the models that are already deployed and running. Replacing existing black-box models in production with interpretable models requires re-training the whole model from scratch, which

can be resource and time consuming.

We propose two novel approaches to improving the explainability of black-box models, which we call *pre-hoc explainability* and *co-hoc explainability*. Our approach aims to incorporate explanations derived from an inherently interpretable white-box model into the original model's learning stage without compromising its high prediction accuracy. Unlike post-hoc explanations, our approach does not rely on input perturbation or secondary model learning, thus avoiding the potential pitfalls of surrogate modeling. Instead, we leverage the insights provided by a white-box model to guide the training of the black-box model in a way that preserves its accuracy while enhancing its global interpretability. Our approach outperforms traditional black-box and white-box models on several benchmark datasets and offers a promising direction to make machine learning models more transparent and trustworthy.

### 1.1 Problem Statement

Let $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{\mathcal{N}} \subset \mathcal{Z}$ be a sample from a distribution $\mathcal{D}$ in a domain $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the instance, and $\mathcal{Y}$ is the label set. We learn a differentiable predictive function $f \in \mathcal{F} : \mathcal{X} \to \mathcal{Y}$ together with a transparent function $g \in \mathcal{G} : \mathcal{X} \to \mathcal{Y}$ defined over a functional class $\mathcal{G}$. We refer to functions $f$ and $g$ as the predictor and the explainer, respectively, throughout the paper. $\mathcal{G}$ is strictly constrained to be an inherently explainable functional set, such as a set of linear functions or decision trees. We assume that we have a distance function $d : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ such that $d(y, \hat{y}) = 0 \longleftrightarrow y = \hat{y}$, which measures the point-wise similarity between two probability distributions in $\mathcal{Y}$ and can be used to optimize $f$ and $g$.

Instead of learning a post-hoc white-box model, our idea is to learn a white-box model that is explainable from the start and then let this explainer model guide the learning of the black-box predictor model. This approach aims to address the limitations of post-hoc explanations, such as potential discrepancies between the explainer and the black-box model [25], and the computational overhead associated with generating explanations after model training [21].

To accomplish this goal, we design two different frameworks: (1) A Pre-Hoc Explainable Predictive Framework, where the white-box model regularizes the black-box model for optimized fidelity, and (2) A Co-hoc Explainable Predictive Framework, where the white-box and black-box models are optimized simultaneously with a shared loss function that

enforces fidelity.

In the Pre-Hoc Explainable Predictive Framework, we first train the explainer model $g$ and then use it to guide the learning of the predictor model $f$. The objective function for training the predictor model includes a fidelity term that minimizes the distance between the predictor's and explainer's outputs, encouraging the predictor to mimic the explainer's behavior. This approach ensures that the predictor model is regularized by the explainer model, leading to improved interpretability.

In the Co-hoc Explainablity Framework, we jointly optimize the predictor model $f$ and the explainer model $g$ during training. The shared loss function consists of both models' standard supervised learning objective (e.g., cross-entropy loss) and a fidelity term that minimizes the distance between their outputs. By simultaneously training both models, we encourage the predictor to learn from the explainer and the explainer to adapt to the predictor, resulting in a more coherent and interpretable system.

Our proposed frameworks differ from existing approaches in several aspects. First, we integrate interpretability directly into the model training process rather than relying on post-hoc explanations. Second, we use a transparent white-box model to guide the learning of the black-box model, ensuring that the explanations are faithful to the predictor's behavior. Finally, our frameworks are model-agnostic and can be applied to any differentiable predictor and explainer models.

In the following chapters, we will formally define the objectives and algorithms for both the Pre-Hoc and Co-hoc Explainablity Frameworks and demonstrate their effectiveness through empirical studies on various datasets and model architectures.

### 1.2 Research Contributions

We propose two novel approaches to improve the explainability of black-box models: pre-hoc explainability for global interpretability and co-hoc explainability for local interpretability. These approaches integrate explainability directly into the training process, ensuring faithful and consistent explanations without requiring additional post-hoc computations. We then extend our pre-hoc framework to generate instance-specific explanations by incorporating the Jensen-Shannon (JS) divergence as a regularization term, capturing the local behavior of the black-box model. Our methods avoid the pitfalls of surrogate modeling, such as instability and unfaithfulness, making them more scalable, robust, and

reliable compared to post-hoc techniques like LIME. We demonstrate the effectiveness of our approaches on real-world datasets, showing improved fidelity and comparable accuracy to traditional black-box models, while providing more faithful and consistent explanations at a lower computational cost than LIME. Finally, we provide a theoretical analysis, proving that our methods balance accuracy and interpretability through regularized learning, and derive bounds on their generalization error.

To summarize, we list the main contributions of our dissertation, below:

1. We propose two novel approaches to improving the explainability of black-box models, called *pre-hoc explainability* and *co-hoc explainability*, which leverage the insights provided by an inherently interpretable white-box model to guide the training of the black-box model in a way that preserves its accuracy while enhancing its global interpretability. These approaches integrate explainability directly into the training process, ensuring that the explanations are faithful to the model's behavior and do not require additional post-hoc computations.

2. We extend our pre-hoc explainability framework to provide local explanations by incorporating the Jensen-Shannon (JS) divergence [26] as a regularization term in the loss function. This allows our method to generate instance-specific explanations that capture the local behavior of the black-box model, similar to post-hoc methods such as LIME [17]. However, unlike LIME, our approach integrates local explainability directly into the training process, ensuring that the explanations are faithful to the model's behavior and consistent with the model's predictions for similar instances.

3. Unlike post-hoc explanations, our approaches do not rely on random input perturbation and post-secondary model learning, thus avoiding the potential pitfalls of surrogate modeling, such as instability and unfaithfulness [21, 27]. This makes them more scalable, robust, and reliable in practice. By incorporating global and local explainability through an interpretable white-box model and the JS divergence, our methods can generate more accurate and stable explanations compared to LIME.

4. Enhancing the accuracy of white-box models through the co-hoc learning framework. The white-box model, which is learned for the purpose of explaining the black-box predictor, achieves significantly higher prediction accuracy after the co-hoc learning

process. This finding highlights the potential of the co-hoc in-training approach to improve the performance of white-box models, which are essential and required in certain high-risk and regulated application tasks in healthcare and legal decision-making.

5. We demonstrate the effectiveness of our approaches on three benchmark datasets from diverse domains (credit risk scoring, movie recommendations, and robotic grasper failure detection), showing that they outperform traditional black-box models in terms of fidelity while maintaining comparable accuracy. We also compare our methods with the LIME post-hoc explainability technique [17] in terms of the quality and stability of the generated explanations, as well as the computational cost associated with generating explanations. Our results indicate that our approaches provide more faithful and consistent explanations across different instances, as measured by the proposed metrics of global fidelity, local fidelity, and stability. Furthermore, our methods exhibit significantly lower computational costs compared to LIME, as they do not require additional extensive post-hoc computations for training to generate explanations.

6. We provide a theoretical analysis of our approaches, showing that they can be seen as a form of regularized learning that balances the trade-off between accuracy and interpretability. The incorporation of the interpretable white-box model and the JS divergence as regularization terms in the loss function encourages the black-box model to learn a decision boundary that is more aligned with the interpretable model, leading to improved global and local explainability.

### 1.3 Document Organization

The remainder of this dissertation is organized as follows.

Chapter 2 reviews the explainable artificial intelligence (XAI) literature, covering inherently interpretable models, post-hoc explainability techniques, model-specific explainability, and existing in-training explainability approaches. The chapter also discusses the various types of explanations and their implications for interpretability and user trust.

Chapter 3 introduces our proposed explainable machine learning frameworks, pre-hoc and co-hoc explainability, which integrate interpretability directly into the training process of black-box models. We define the problem formulation, present the fidelity objective function, and describe the details of the implementation of our frameworks. We also present

experimental results on real-world datasets, by doing an ablation study, we compare our methods with traditional black-box models and the post-hoc explainability technique LIME in terms of accuracy and fidelity.

Chapter 4 extends our pre-hoc and co-hoc explainability frameworks to incorporate local interpretability by integrating the Jensen-Shannon (JS) divergence as a regularization term. We compare our approach with LIME in terms of the quality and stability of the generated local explanations, as well as the computational costs of both methods.

Finally, Chapter 5 concludes the dissertation by summarizing our contributions, discussing the limitations of our work, and outlining potential future research directions.

# CHAPTER 2

# LITERATURE REVIEW

In this chapter, we review the landscape of explainable artificial intelligence (XAI). Our exploration begins with inherently interpretable models, where we discuss foundational models such as linear regressions and decision trees, known for their transparency and straightforwardness. This leads us next to Post hoc Explainability, a domain where explanations are generated *after* model training, to explain the decision-making of more complex, often opaque models. We further examine Model-Specific Explainability (Explainability by Design), which involves designing models with built-in explainability features, ensuring clarity from the outset. Our attention then shifts to Existing In-Training Explainability Techniques, exploring innovative methods that integrate explainability directly into the model training process. Finally, in Explanation Types, we examine the various forms explanations can take, from visual to textual, and their impact on interpretability and user trust. This chapter aims to provide a comprehensive overview of the current state, challenges, and advances in the field of XAI, establishing a solid foundation for understanding the nuances and significance of explainability in machine learning.

## 2.1 Inherently Interpretable Models

### 2.1.1 Linear Models

Linear models are a cornerstone of statistical modeling and machine learning, prized for their simplicity, interpretability, and foundational role in various predictive tasks. This category encompasses Linear Regression, Logistic Regression, and Generalized Linear Models (GLMs).

Linear Regression is one of the most basic and widely used statistical techniques [28]. It models the relationship between a dependent variable and one or more independent variables using a linear equation. Its simplicity in interpretation and robustness in prediction makes it a preferred choice for many regression tasks.

Logistic regression, on the other hand, is used for binary classification problems [29]. It models the probability of a binary response based on one or more predictor variables. The outputs are transformed using the logistic function, ensuring that the predictions fall between 0 and 1, making it suitable for estimating the probabilities.

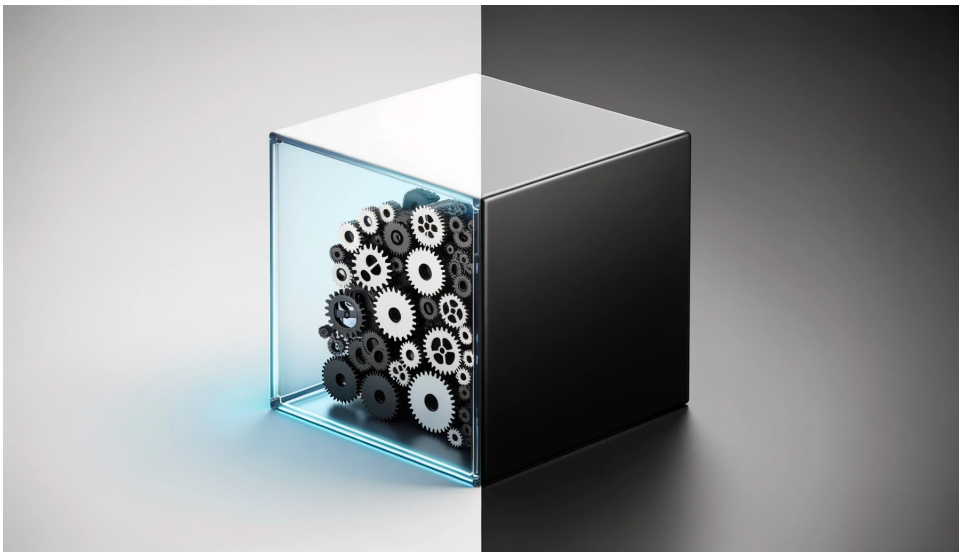


Figure 2.1: The image illustrates a white-box model, represented by a semi-transparent box with visible gears inside on the left, and a black-box model, depicted as a label-free opaque black box on the right. Source: generated using DALL-E.

Generalized Linear Models (GLMs) extend the concept of linear regression by allowing response variables that have error distribution models other than a normal distribution [30]. They unify various other statistical models, including logistic and Poisson regression [31], under a single framework, making them extremely flexible for modeling different types of data.

Linear models are extensively used in fields ranging from economics to biology for their ease of implementation and interpretation. They are particularly valuable when it is important to understand the role and significance of individual predictor features in the model. However, linear models have limitations, especially in handling non-linear relationships, interactions between features, and complex data structures [32]. They are also prone to overfitting with high-dimensional data and may require regularization techniques like Ridge [33] or Lasso [34] to mitigate this issue. Recent advances in linear models focus on improving their interpretability and robustness, especially in high-dimensional settings. Methods such as Elastic-Net regularization combine Lasso and Ridge to improve model

performance and interpretability [35]. Despite the advent of more complex models, linear models continue to be a fundamental tool in statistical modeling and machine learning, offering a blend of simplicity, interpretability, and effectiveness.

### 2.1.2 Decision Trees

Decision Trees [36] constitute one of the most fundamental and easily interpretable family of machine learning algorithms. They are used extensively for both classification and regression tasks. The core principle behind decision trees is to model the decision-making process through a tree-like structure of rules and conditions, visually and logically representing how input features lead to a decision or prediction.

A decision tree consists of nodes, branches, and leaves. Each internal node represents a test on an attribute, each branch corresponds to the outcome of the test, and each leaf node represents a class label or a decision. The paths from root to leaf represent the classification rules. The most common algorithms to build decision trees include ID3 [36], C4.5 [37], and CART [38]. These algorithms differ in how they select which attribute to split on at each step (e.g., using measures like information gain).

Decision trees are widely used because of their simplicity and interpretability. They are particularly useful in domains where it is essential to understand the reasoning behind a model's decision, such as in finance for credit scoring or in medicine for diagnostic purposes [39]. The ability to visually represent the decision-making process allows users to easily grasp how the model comes to a conclusion, fostering trust and transparency. However, decision trees are not without limitations. One major issue is their tendency to overfit the training data, especially when dealing with complex trees. This overfitting can be mitigated through techniques such as pruning (reducing the size of the tree), setting a minimum number of samples per leaf, or limiting the depth of the tree. Another challenge is that small changes in the data can lead to significantly different trees, which can affect the stability of the model [40]. Ensemble methods like Random Forests [41] and Gradient Boosting Trees [42] have been developed to address some of these challenges. These methods combine multiple decision trees to improve performance and robustness [43]. "Random Forests," for example, creates a 'forest' of decision trees trained on random subsets of the data and features and then aggregates their predictions. This approach not only improves the accuracy of the prediction but also helps to reduce overfitting.

Decision trees remain a popular choice for many applications due to their intuitive nature and ease of interpretation. While they may not always be the most powerful or accurate models, their transparency and simplicity make them an essential tool in the domain of interpretable machine learning.

## 2.2 Post-hoc Explainability

Post-hoc explainability approaches are widely used to generate explanations for the predictions made by a trained black-box model. However, because the training and explanation generation phases are decoupled (e.g. LIME), they create the risk of having explanations that are a result of some artifacts learned by the model instead of actual knowledge from the data [44]. Post-hoc explanation techniques that rely on input perturbations, such as LIME and SHAP, are not reliable [21].

Model-agnostic classifiers produce explanations without changing the model either locally in a few instances or globally across all instances. Explanations can be local or global and textual and/or visual. One of the state-of-the-art methods is LIME [17]. LIME is an algorithm that, in a faithful manner, approximates a regressor or classifier; it is used to discover interpretable models. Another commonly used library is SHAP (SHapley Additive exPlanations) [18] is a system integration framework for interpreting predictions. It rates each function according to its value for the prediction. Its innovative components include discovering a new class of additive function importance measurements and theoretical studies suggesting a notable solution with a range of attractive properties in this class.

Model-agnostic methods can validate and analyze a model through various machine learning algorithms (post hoc). These automated processes are usually performed by evaluating the features and their subsequent output. They cannot access the internal model state for these approaches to operate, including weights or structural details.

### 2.2.1 LIME (Local Interpretable Model-Agnostic Explanations)

Local Interpretable Model-Agnostic Explanations (LIME) emerged as a significant development in the field of machine learning explainability. Introduced by Ribeiro et al. in their landmark paper [17], LIME addresses the challenge of interpreting complex black-box models. It operates on a simple yet profound premise: While it may be challenging to

explain an entire model, it is possible to approximate and explain individual predictions.

In particular, LIME, one of the most popular post-hoc explanation systems, solves the following optimization problem:

$$e(x, f) := \arg\min F\left(f, g, N_x\right) + \Omega(g) \tag{2.1}$$

where $\Omega(g)$ stands for an additive regularizer that encourages certain desirable properties of the explanations (e.g., sparsity).

LIME generates explanations by perturbing the input data and observing the changes in the model's predictions. This process involves creating a new dataset consisting of perturbed samples and the corresponding predictions from the black-box model. LIME then trains an interpretable surrogate model, such as a linear model, on this new data set; see Algorithm 0. The interpretable model is designed to be locally faithful, meaning it accurately represents the black-box model behavior in the approximation of the instance being explained.

**Algorithm 2.1** Sparse Linear Explanations using LIME

**Require:** Classifier $f$, Number of samples $N$
**Require:** Instance $x$, and its interpretable version $x'$
**Require:** Similarity kernel $\pi_x$, Length of explanation $K$
  $Z \leftarrow \{\}$
  **for** $i \leftarrow 1$ to $N$ **do**
    $z_i' \leftarrow$ `sample_around`$(x')$
    $Z \leftarrow Z \cup \{(z_i', f(z_i), \pi_x(z_i'))\}$
  **end for**
  $w \leftarrow$ `K-Lasso`$(Z, K)$ ▷ with $z_i'$ as features, $f(z)$ as target
  **return** $w$

LIME has been widely adopted because of its flexibility and model-agnostic nature, allowing it to be used with any machine learning model. It has been particularly impactful in fields where understanding model predictions is crucial, such as healthcare and finance [45]. For example, in medical diagnostics, LIME can help clinicians understand why a model recommends a particular treatment, thus improving trust in AI-driven decision-making processes.

Despite its popularity, LIME is not without criticism. Researchers have pointed out that LIME's explanations can sometimes be unstable, meaning that slight changes in the input can lead to significantly different explanations [27]. Furthermore, LIME's reliance on local linear approximations may not always capture the complexities of certain models,

especially in regions of the input space where the model's behavior is highly non-linear [21]. In response to these critiques, there have been efforts to improve the stability and reliability of LIME. For example, some researchers have proposed enhancements to the sampling methodology to ensure more robust and consistent explanations [46]. Furthermore, there is ongoing research on the development of methodologies that can provide more global insight while maintaining the local fidelity that LIME offers [47].

### 2.2.2 SHAP (SHapley Additive exPlanations)

SHapley Additive exPlanations (SHAP) is an important approach in the field of explainable artificial intelligence, introduced by Lundberg and Lee in 2017 [48]. SHAP is based on the concept of Shapley values from cooperative game theory, providing a unified measure of feature importance.

The theoretical basis of SHAP lies in the Shapley value, a method of game theory that fairly allocates the payout of a game to its players [49]. In the context of machine learning, SHAP treats each feature value of an instance as a 'player' in a game where the 'payout' is the prediction made by the model. SHAP calculates the contribution of each feature to the difference between the actual prediction and the average prediction over the dataset. SHAP's model-agnostic nature allows it to be used with any machine learning model, making it highly versatile. It has been particularly influential in sectors such as healthcare and finance, where understanding the impact of individual variables is crucial to decision making [50]. SHAP values provide a detailed and interpretable overview of feature contributions, offering local and global explanations. For individual predictions, SHAP can highlight which features were most influential, and at the global level, it can indicate how much each feature generally impacts the model's output.

Although SHAP offers deep insight, it can be computationally expensive, especially for models with a large number of features or complex architectures [51]. There are also concerns about the interpretability of SHAP values in highly interactive or correlated feature spaces, as contributions may not be entirely intuitive. In response to computational challenges, various optimizations, and approximations have been proposed to make SHAP more scalable [52]. Researchers continue to explore ways to enhance SHAP's efficiency and interpretability, making it more applicable to a broader range of models and datasets.

## 2.3 Model-Specific Explainability (Explainability by Design)

Model-specific interpretation tools are restricted to narrow model classes. One of the ways to achieve explainability is through architectural adjustments. They make the model layout more comprehensible by modifying the model architecture. Regularization approaches are not only used to maximize predictive precision; they can also be used to improve the explainability of AI models [53] [54] [55] [56].

Abdollahi and Nasraoui [53] proposed an explainable matrix factorization approach for collaborative filtering. They introduced a constrained matrix factorization model that incorporates user and item biases, as well as user and item factors. The model architecture is designed to provide interpretable recommendations by learning latent factors that correspond to explicit user preferences and item characteristics. The regularization term in the objective function encourages the model to learn sparse and meaningful latent factors, enhancing the explainability of the recommendations.

Ras et al. [54] developed an explainable 3D convolutional neural network (CNN) to analyze time series data. They modified the architecture of a 3D CNN by incorporating a temporal attention mechanism and a feature attention mechanism. The temporal attention mechanism allows the model to focus on the most relevant time steps, while the feature attention mechanism helps identify the most informative features. By designing the model architecture to include these attention mechanisms, the authors improved the explainability of the 3D CNN, enabling users to understand which time steps and features contribute most to the model's predictions.

Fauvel et al. [55] introduced XCM, an explainable convolutional neural network for multivariate time series classification. They designed the model architecture to include a temporal convolution block, which learns temporal patterns, and a feature convolution block, which learns feature dependencies. The model also incorporates a temporal attention mechanism and a feature attention mechanism, similar to the approach by Ras et al. [54]. The XCM architecture enhances the explainability of the model by providing insights into the temporal patterns and feature dependencies that drive the model's predictions.

Miao et al. [56] proposed an interpretable and generalizable graph learning framework based on a stochastic attention mechanism. They designed a graph neural network architecture that incorporates a stochastic attention mechanism, which learns to assign attention

weights to different nodes and edges in the graph. The stochastic nature of the attention mechanism allows the model to explore different subgraphs and capture the uncertainty in the attention weights. The authors also introduced a regularization term in the objective function to encourage the model to learn sparse and interpretable attention weights. By designing the model architecture and the objective function to promote interpretability, the framework provides insights into the graph structures and node relationships that influence the model's predictions.

In addition to architectural modifications, regularization techniques can also be used to improve the explainability of AI models. L1 regularization, also known as Lasso regularization, is a commonly used technique that encourages sparsity in the model parameters [57]. By inducing sparsity, L1 regularization helps identify the most relevant features of the model, making it more interpretable. L2 regularization, also known as Ridge regularization, is another technique that can improve the stability and generalization of model performance [58]. Although L2 regularization does not directly promote sparsity, it can help prevent overfitting and improve the model's robustness, which can indirectly contribute to its explainability.

Other regularization techniques, such as dropout [59] and early stopping [60], can also be used to improve the interpretability of neural AI models. Dropout is a regularization technique that randomly drops out neurons during training, preventing the model from relying too heavily on specific features or neurons. By encouraging the model to learn more robust and generalizable representations, dropout can improve the model's interpretability. Early stopping is a technique that halts the training process when the model's performance on a validation set starts to degrade. Early stopping can help the model learn more meaningful and interpretable patterns in the data by preventing overfitting.

In summary, model-specific explainability can be achieved through architectural adjustments and regularization techniques. Researchers can improve the explainability of AI models by designing the model architecture to incorporate interpretable components, such as attention mechanisms and sparsity-inducing regularizers. Regularization techniques, such as L1 and L2 regularization, dropout, and early stopping, can further enhance the interpretability of the model by promoting sparsity, robustness, and generalization. As the field of explainable AI continues to evolve, model-specific approaches will likely play a crucial role in developing transparent and interpretable AI systems tailored to specific domains and applications.

## 2.4 Existing In-Training Explainability Techniques

The majority of the existing work on explainable AI has focused on either developing post hoc explanation methods for black-box models or building models that are explainable by design. Post-hoc techniques analyze trained models to provide explanations for individual predictions [18] [17] [61], either with model-specific methods based on input perturbations or model-agnostic explainer models. However, post hoc approaches have been criticized for potential discrepancies between the explainer and the black-box model [21] [44]. On the other hand, model-specific explainability has its own limitations as it requires individual methods and implementations for each different black-box model.

On the contrary, research on improving explainability through model training is more limited. Only a few methods have explored the use of interpretable models to directly guide black-box training for higher explainability, starting with methods such as [53] for recommendation models. Using tree regularization [62] to train deep time-series models, with the aim of human-simulability [63]. Other works proposed training models with latent explainability, but they still rely on post hoc explanations [64] [65]. An alternative approach is to use a game-theoretic approach between predictor and explainer [66], [67]. Using a cooperative game, they optimize the explainer for locality, specifically for sequential data. [68] used a regularization approach to push black-box models toward relying more on interpretable features, but their explanations remain post-hoc, specifically optimized for LIME's neighborhood-based fidelity, which has to be computed *at prediction time*. In fact, their goal is to improve the *quality* of *post-hoc* explanations of the model, thus they do not attempt to solve the same problem as ours, as we do not rely on post-hoc explanations. Another line of work was designed to learn the latent concept-based explanations implicitly during training, which eliminates the requirement of post-hoc explanation generation techniques [69]. Because the concepts must be learned using either external annotation or self-supervision, e.g. using auto-encoders from the input features, this approach is limited to special input types like images or domains with available external supervision.

In general, research on improving model explainability highlights the need for further work on optimization *during training*, and model-agnostic methods to improve global explainability. Our approach addresses this need by directly injecting global interpretability into black-box learning, *at training time*, through an interpretable explainer model that does

*not* require additional post hoc computation at prediction time.

## 2.5 Explanation Types

In the domain of XAI, explanations can be categorized into various types, each offering a unique lens to interpret machine learning models. Understanding these types is crucial for selecting appropriate explanation methods depending on the context and requirements [70].

The explanation types in XAI offer diverse perspectives on interpreting machine learning models, catering to different contexts and requirements. Rule-based explanations provide human-readable rules [71], while feature-based explanations quantify the impact of individual features [17]. Concept-based explanations align model behavior with human-understandable concepts [72], and instance-based explanations highlight influential data instances [9]. Local explanations offer insights into individual predictions [48], whereas global explanations describe the model's overall behavior [47]. Multi-model explanations compare explanations across different models, providing a comparative view [73].

Choosing the appropriate explanation type depends on various factors, such as the target audience, the complexity of the model, and the specific application domain. For example, in healthcare, concept-based explanations may be more suitable for communication with medical professionals, while rule-based explanations may be more appropriate for patient-facing applications [74]. In finance, feature-based explanations can help identify key risk factors, while instance-based explanations can assist in detecting anomalous transactions [75]. Moreover, the different explanation types can be combined to provide a more comprehensive understanding of the model behavior. For instance, local explanations can be aggregated to generate global insights, while multi-model explanations can incorporate both feature-based and concept-based explanations to offer a holistic view of the models' decision-making processes [76].

As the field of XAI continues to evolve, new types of explanation and techniques are likely to emerge, addressing the limitations of existing approaches and catering to the ever-growing complexity of machine learning models. Researchers and practitioners should stay abreast of these developments and adopt a multifaceted approach to explainability, leveraging the strengths of different explanation types to ensure transparency, accountability, and trust in AI systems [77]. Understanding the various explanation types in XAI is crucial for

developing effective and meaningful explanations of machine learning models. By carefully considering the strengths and limitations of each type of explanation and adapting them to the specific context and requirements, we can create more transparent, interpretable, and trustworthy AI systems that benefit both developers and end-users alike [78].

### 2.5.1 Rule-based Explanations

Rule-based explanations provide information by simplifying model decisions into human-readable rules. Decision trees are a classic example where decisions can be broken down into a series of if-then rules [36]. This type of explanation is highly intuitive, making it easier for users to grasp the logic behind predictions. Rule-based explanations are particularly useful in domains where transparency and interpretability are crucial, such as healthcare, finance, and legal systems [79, 80]. One of the main advantages of rule-based explanations is their simplicity and comprehensibility. They allow users to understand the decision-making process of a model without requiring deep technical knowledge. This is especially important in scenarios where the model's predictions have significant consequences, and users need to trust and justify the model's decisions [17]. However, rule-based explanations also have some limitations. As the complexity of the model increases, the number of rules required to explain its behavior may become large and unwieldy, making it difficult for users to understand the entire decision-making process [71]. Additionally, rule-based explanations may not capture the full nuance and complexity of the model's behavior, especially in cases where the model relies on non-linear or high-dimensional relationships between features [81]. Despite these limitations, rule-based explanations can be combined with other explanation types, such as feature-based or local explanations, to provide a more comprehensive understanding of the model's behavior. Furthermore, recent advances in rule-based explanations, such as hierarchical rule-based explanations [82] and rule extraction techniques [83], have aimed to address some of the limitations and improve the scalability and fidelity of rule-based explanations.

### 2.5.2 Feature-based Explanations

Feature-based explanations focus on the contribution of individual features to the model's prediction. Techniques like SHAP (SHapley Additive exPlanations) [48] and LIME (Local Interpretable Model-agnostic Explanations) [17] fall into this category, quantifying

the impact of each feature on the model's output. These explanations help in understanding the relative importance of features and how they influence the model's decisions.

SHAP is a game-theoretic approach that assigns each feature an importance value, known as the Shapley value, which represents the feature's contribution to the model's prediction [48]. The Shapley values are computed by considering all possible feature combinations and measuring each feature's impact on the model output. SHAP provides both local and global explanations, allowing users to understand the importance of features for individual predictions as well as the overall model behavior.

LIME, on the other hand, is a local explanation technique that approximates the model behavior around a specific instance by training a simple, interpretable model (e.g., a linear model) on perturbed samples in the surroundings of the instance [17]. The coefficients of the interpretable model serve as a measure of the importance of each feature for the particular instance. LIME is model-agnostic, meaning it can be applied to any black-box model, making it a versatile tool for generating feature-based explanations.

Feature-based explanations have several advantages. They provide a quantitative measure of the importance of each feature, allowing users to identify the most influential factors in the model's decision-making process. This information can be valuable for feature selection, model debugging, and identifying potential biases or errors in the model [76]. Furthermore, feature-based explanations can be easily visualized using techniques such as feature importance plots or heatmaps, making them accessible to a wide range of users [84]. However, feature-based explanations also have some limitations. They may not capture complex interactions between features or non-linear relationships, as they often assume feature independence and additive contributions [85]. Furthermore, the interpretation of feature importance scores can be challenging, especially when dealing with high-dimensional or correlated features [81].

Despite these limitations, feature-based explanations remain a popular and effective approach to interpreting machine learning models. Recent advances, such as SHAP interaction values [52] and anchors [47], have aimed to address some of the limitations and provide more comprehensive and robust feature-based explanations.

In summary, feature-based explanations offer a quantitative approach to understanding the importance of individual features in a model's decision-making process. Techniques such as SHAP and LIME have become widely adopted in the XAI community, providing valu-

able insights into model behavior and helping to promote transparency and interpretability in AI systems.

### 2.5.3 Concept-based Explanations

Concept-based explanations aim to express model decisions using high-level concepts that are more understandable to humans. This approach is particularly useful in complex domains such as image recognition, where explanations are provided in terms of visual or semantic concepts [72]. Concept-based explanations bridge the gap between the model's internal representation and the user's domain knowledge by aligning model behavior with human-understandable concepts. One of the techniques to generate concept-based explanations is Concept Activation Vectors (CAVs) [72]. CAVs are linear classifiers that are trained to distinguish between instances that contain a specific concept and those that do not. The concept can be defined by a set of examples or by a user-provided label. Once trained, CAVs can be used to measure the sensitivity of a model's predictions to the presence or absence of the concept, providing a quantitative measure of the concept's importance. Another approach to concept-based explanations is Automatic Concept-based Explanations (ACE) [86]. ACE automatically extracts concepts from the model's internal representation using unsupervised learning techniques, such as clustering or principal component analysis. The extracted concepts are then used to generate explanations by identifying the concepts that are most relevant to a particular instance or decision.

Concept-based explanations provide explanations in terms that are more easily understandable to users because they relate to high-level concepts rather than low-level features. This can help users build trust in model decisions and identify potential errors or biases [72]. In addition, concept-based explanations can be used to assess model alignment with human values and expectations, which is crucial in domains such as healthcare and criminal justice [87]. However, concept-based explanations also have some limitations. The quality of the explanations depends on the choice of concepts and the ability to accurately extract them from the model's internal representation. If the concepts are not well aligned with the model behavior or the user domain knowledge, the explanations may be misleading or incomplete [88]. Furthermore, concept-based explanations may not capture the full complexity of the model decision-making process, as they focus on a limited set of high-level concepts [81].

Concept-based explanations are particularly used in domains where human-understandable concepts are readily available. Recent advances such as Concept Bottleneck Models [89] and Concept Whitening [90], have aimed to improve the quality and robustness of concept-based explanations by incorporating concept information directly into the model's architecture. Concept-based explanations offer a way to interpret machine learning models using high-level concepts that are more easily understandable by humans. By bridging the gap between the model internal representation and user domain knowledge, concept-based explanations can help to promote trust, transparency, and accountability in AI systems.

### 2.5.4 Instance-based Explanations

Instance-based explanations provide insights by highlighting specific data instances that are influential in the model's learning process. For example, influential instances used in training can be identified to explain model behavior [9]. These explanations help users understand how the model has learned from specific examples and how those examples have shaped its decision-making process. One major technique for generating instance-based explanations is influence functions [9]. Influence functions measure the effect of removing or perturbing individual training instances on the model's predictions. By identifying the most influential instances, influence functions can provide insights into the model's behavior and help users understand why the model makes certain predictions. Another approach to instance-based explanations is prototype selection [91]. Prototype selection involves identifying a subset of the training data that is most representative of the model's decision boundary. These prototypes serve as a condensed version of the training data, providing a more interpretable representation of the model's behavior. Users can examine the prototypes to understand the key patterns and features that the model has learned.

Instance-based explanations provide a concrete and tangible way to understand the model behavior, as users can directly examine the instances that have influenced the model's predictions. This can be particularly useful in applications such as natural language processing or image classification [9, 92]. In addition, instance-based explanations can help users identify potential biases or errors in the training data, as they highlight the instances that have had the greatest impact on model behavior. However, instance-based explanations also have some limitations. Interpreting influential instances can be challenging, as they may not always provide a clear or complete picture of the model behavior [81]. Additionally,

the selection of influential instances or prototypes can be sensitive to the choice of metrics or algorithms used, which can lead to different explanations depending on the approach taken [93]. Despite these limitations, instance-based explanations remain a valuable tool in the XAI toolkit. Recent advancements, such as counterfactual explanations [94] and adversarial examples [95], have aimed to provide more robust and comprehensive instance-based explanations by exploring the model's behavior in the vicinity of specific instances. Instance-based explanations offer a way to understand machine learning models by highlighting the specific data instances that have influenced the model's behavior. By examining influential instances or prototypes, users can gain insight into the patterns and features that the model has learned, helping to promote transparency and interpretability in AI systems.

### 2.5.5 Local Explanations

Local explanations offer insights into the model's behavior for individual predictions. They explain why a model made a specific decision for a particular instance, which is crucial in applications where individual decisions need to be justified [17]. Local explanations provide a fine-grained understanding of the model's decision-making process, allowing users to examine the factors that contributed to a particular prediction.

One of the most widely used techniques for generating local explanations is LIME (Local Interpretable Model-agnostic Explanations) [17]. LIME works by approximating the model's behavior locally for a specific instance using a simple, interpretable model (e.g., a linear model). The interpretable model is trained on perturbed samples around the instance, and the coefficients of the model serve as a measure of the importance of each feature for the particular prediction. LIME is model-agnostic, which can be applied to any black-box model, making it a versatile tool for generating local explanations. Another approach to local explanations is Shapley Additive Explanations (SHAP) [48]. SHAP is based on the concept of Shapley values from cooperative game theory, which assigns each feature an importance value based on its contribution to the model's prediction. SHAP provides local explanations by computing the Shapley values for each feature for a specific instance, allowing users to understand the impact of each feature on the particular prediction.

Local explanations have several advantages. They provide a detailed understanding of the model behavior for individual instances, which is crucial in domains where the consequences of individual decisions are significant, such as healthcare and criminal justice [87].

Moreover, local explanations can help users identify potential biases or errors in the model predictions, as they highlight the factors that contributed to a particular decision. However, local explanations also have some limitations. They may not provide a complete picture of the model's overall behavior, as they focus on individual instances rather than global patterns [81]. Furthermore, the interpretation of local explanations can be challenging, as they may be sensitive to the choice of perturbation strategies or interpretable models used [27]. Despite these limitations, local explanations remain an essential tool in the XAI toolkit. Recent advancements, such as anchors [47] and counterfactual explanations [94], have aimed to provide more robust and comprehensive local explanations by exploring the model's behavior in the surroundings of specific instances.

In summary, local explanations offer a way to understand the model's behavior for individual predictions, providing a fine-grained understanding of the factors that contributed to a particular decision. Techniques such as LIME and SHAP have become widely adopted in the XAI community, helping to promote transparency and interpretability in AI systems.

### 2.5.6 Global Explanations

Global explanations provide an overview of the model's general behavior across all instances. They aim to describe the general logic and patterns learned by the model, which is essential for understanding the behavior of the model on a larger scale [71]. Global explanations offer a high-level understanding of the model's decision-making process, allowing users to grasp the key factors and relationships that drive the model's predictions.

One approach to generating global explanations is through the use of interpretable models, such as decision trees [36] or rule-based systems [79]. These models are inherently interpretable, as they provide a transparent representation of the decision-making process in the form of a tree or a set of rules. By training an interpretable model on the same data as the black-box model, users can understand the model's behavior globally. Another approach to global explanations is through the use of feature importance methods, such as permutation importance [43] or global surrogate models [17]. Permutation importance measures the impact of each feature on the model's performance by randomly shuffling the values of the feature and observing the resulting change in the model's accuracy. Global surrogate models, on the other hand, involve training a simple, interpretable model (e.g., a linear model) on the predictions of the black-box model, providing a global approximation

of the model's behavior.

Global explanations provide a comprehensive understanding of the model's behavior, allowing users to identify the key factors and relationships that drive the model's predictions. This can be particularly useful in domains where the overall behavior of the model is of interest, such as in policy making or strategic decision making [87]. Moreover, global explanations can help users identify potential biases or limitations in the model's behavior, as they provide a broad overview of the model's decision-making process. However, global explanations also have some limitations. They may not capture the full complexity of the model's behavior, as they provide a simplified representation of the decision-making process [81]. Additionally, global explanations may not be sufficient for understanding the model's behavior in specific instances, as they focus on overall patterns rather than individual predictions. Despite these limitations, global explanations remain an essential tool in the XAI toolkit. Recent advances, such as concept-based explanations [72] and global feature importance methods [96], have aimed to provide more comprehensive and interpretable global explanations by incorporating domain knowledge and local explanations. Global explanations offer a way to understand the overall behavior of machine learning models, providing a high-level understanding of the key factors and relationships that drive the model's predictions. By offering a comprehensive view of the model's decision-making process, global explanations can help to promote transparency and interpretability in AI systems.

### 2.5.7 Multi-Model Explanations

Multi-model explanations involve generating explanations across different models. This approach provides a comparative view, helping users understand how different models process information and make decisions [97]. By examining similarities and differences between the explanations generated by multiple models, users can gain insight into the robustness and generalizability of the model behavior. One approach to multi-model explanations is through the use of model comparison techniques, such as model distillation [98] or model compression [99]. These techniques involve training a simpler, more interpretable model to mimic the behavior of a complex, black-box model. By comparing the explanations generated by the simpler model with those of the black-box model, users can gain insight into the key factors and relationship captured by both models. Another approach

to multi-model explanations is through the use of ensemble methods, such as bagging [100] or boosting [101]. Ensemble methods involve combining the predictions of multiple models to improve overall performance and robustness of the system. By generating explanations for each model in the ensemble and comparing them, users can gain insight into the diverse perspectives and strategies employed by the different models.

Multi-model explanations have several advantages. They provide a more comprehensive understanding of the model behavior, as they allow users to examine the similarities and differences between the explanations generated by multiple models. This can be particularly useful in domains where the robustness and generalizability of the model's behavior are of concern, such as in healthcare or finance [102]. Moreover, multi-model explanations can help users identify potential biases or limitations in the individual models, as they provide a comparative view of the models' decision-making processes. Generating and comparing explanations in multiple models can be computationally expensive, particularly when dealing with large-scale or complex models [81]. Additionally, the interpretation of multi-model explanations can be challenging, as users may need to reconcile the potentially diverse or conflicting explanations generated by the different models. Multi-model feature importance methods [96] aimed to providing more comprehensive and interpretable multi-model explanations by incorporating local and global explanations in multiple models. Multi-model explanations offer a way to understand the behavior of machine learning models by comparing the explanations generated by multiple models. By providing a comparative view of the model decision-making processes, multi-model explanations can help to promote transparency, robustness, and generalizability in AI systems.

## 2.6 Explanation Evaluation Metrics

Post-hoc explainability, such as LIME,can be evaluated using three metrics: point fidelity, neighborhood fidelity [68] and stability [27]. Point fidelity measures the agreement between the explainer and predictor for individual instances, while neighborhood fidelity extends this concept to consider the agreement within local neighborhoods around each instance. The stability metric quantifies the variability or change in fidelity scores within the same neighborhood of instance.

#### 2.6.0.1 Point Fidelity

Let $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ be a dataset with $N$ instances, where $\mathbf{x}_i \in \mathbb{R}^d$ is a feature vector and $y_i \in \{0, 1\}$ is the corresponding binary label. Let $f$ be a black-box model predictor and $g$ be a white-box model explainer (e.g., LIME) that generates explanations.

The point fidelity metric [17, 68] for instance $\mathbf{x}_i$ is defined as:

$$\text{PointFidelity}(\mathbf{x}_i) = \mathbb{1}(\hat{y}_{f,i} = \hat{y}_{lime,i}), \tag{2.2}$$

where $\hat{y}_{f,i}$ is the predicted label of model $f$ for instance $\mathbf{x}_i$, $\hat{y}_{lime,i}$ is the predicted label of model $g$ explainer for example $\mathbf{x}_i$ and $\mathbb{1}(\cdot)$ is the indicator function.

The average point fidelity score across all instances is given by:

$$\text{AvgPointFidelity} = \frac{1}{N} \sum_{i=1}^{N} \text{PointFidelity}(\mathbf{x}_i) \tag{2.3}$$

Point Fidelity evaluates the agreement between the explanations and predictions of different models for individual instances in a dataset. It measures how well the explanations generated by an interpretable model, such as LIME, align with the predictions made by a black-box model, a white-box model, and the proposed explainable model. Local fidelity scores quantify the consistency between the explanations and predictions for each instance, assessing the point-wise agreement between the explanations and predictions by averaging the local fidelity scores across all instances. In contrast, loint fidelity helps to evaluate the quality of explanations at the individual instance level without considering the local neighborhood around each instance.

#### 2.6.0.2 Neighborhood Fidelity

Let $\mathcal{N}_k(\mathbf{x}_i)$ denote the set of $k$-nearest neighbors of instance $\mathbf{x}_i$ in the feature space.

The neighborhood fidelity metric for instance $\mathbf{x}_i$ is defined as [68]:

$$\text{NeighborhoodFidelity}(\mathbf{x}_i) = \frac{1}{k} \sum_{\mathbf{x}_j \in \mathcal{N}_k(\mathbf{x}_i)} \mathbb{1}(\hat{y}_{f,j} = \hat{y}_{lime,j}), \tag{2.4}$$

where $\hat{y}_{j,m}$ is the predicted label of model $f$ for instance $\mathbf{x}_j$ in the neighborhood of $\mathbf{x}_i$.

The average neighborhood fidelity score across all instances is given by:

$$\text{AvgNeighborhoodFidelity} = \frac{1}{N} \sum_{i=1}^{N} \text{NeighborhoodFidelity}(\mathbf{x}_i) \tag{2.5}$$

The Neighborhood Fidelity extends the concept of point fidelity by considering the local neighborhood around each instance when evaluating the agreement between explanations and predictions. It measures how well the explanations generated by an interpretable model, such as LIME, align with the predictions made by different models within the local surroundings of each instance, typically the k-nearest neighbors of each instance in the dataset. Neighborhood fidelity scores quantify the consistency of explanations and predictions in the local region surrounding each instance. By averaging the neighborhood fidelity scores across all instances, the algorithm assesses the agreement between explanations and predictions in the local neighborhoods. Neighborhood fidelity complements point fidelity by considering the coherence and stability of explanations and predictions in the local context of each instance.

In summary, point fidelity and neighborhood fidelity algorithms provide valuable insights into the quality and consistency of explanations generated by interpretable models about the predictions made by different models. Point fidelity focuses on the agreement at the individual instance level, while neighborhood fidelity considers the local context around each instance. By evaluating these metrics, researchers can assess the effectiveness of their proposed explainable models and compare them with existing interpretable techniques, such as LIME. The algorithms help quantify the alignment between explanations and predictions, contributing to developing and validating reliable and trustworthy explainable AI systems.

#### 2.6.0.3 Stability

Stability metric [27] is defined as follows:

Let $\mathcal{F} = \{f_1, f_2, \ldots, f_N\}$ be a set of fidelity scores, where $f_i$ represents the fidelity score for the $i$-th instance or neighborhood. The total variation (TV) of the fidelity scores is defined as:

$$\mathrm{TV}(\mathcal{F}) = \frac{1}{N-1} \sum_{i=1}^{N-1} |f_{i+1} - f_i| \tag{2.6}$$

where $|\cdot|$ denotes the absolute value function, and $N$ is the total number of fidelity scores.

The stability metric quantifies the variability or change in neighborhood fidelity scores in different instances. It measures the sum of absolute differences between consecutive fidelity scores, normalized by the number of instances minus one. A lower total variation

indicates higher stability, suggesting that explanations are more consistent and less prone to sudden changes in neighborhood instances. The intuition behind using the total variation as a stability metric is that stable explanations should exhibit smooth transitions between fidelity scores without drastic fluctuations. Suppose the fidelity scores vary significantly from one instance to another. In that case, it implies that the explanations are sensitive to small changes in the input and may not be reliable or consistent. Incorporating the total variation metric alongside other metrics such as average fidelity scores and standard deviations helps to gain a more comprehensive understanding of the stability and robustness of the explanations generated by different models. A model with **high average fidelity scores and low total variation** is desirable, indicating that the descriptions are accurate and stable in different instances.

To summarize, stability complements other fidelity metrics and provides insight into the robustness and consistency of the explanations across different instances or neighborhoods.

## 2.7 Factorization Machine Model

Factorization machines (FMs) [103] are a medium to train supervised learning models that can be applied to a wide range of prediction tasks, including regression, classification, and ranking. FMs can reliably estimate model parameters under large amounts of sparse data, allowing the model to be trained with very few data points.

Model Equation: The model equation for a factorization machine of degree d = 2 is defined as:

$$\hat{y} = w_0 + \sum_{1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{i,j} x_i x_j \tag{2.7}$$

The factorization machine with pairwise interactions is defines as:

$$\hat{y}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j \tag{2.8}$$

where the model parameters that have to be estimated are:

$$w_0 \in \mathbb{R}, w_i \in \mathbb{R}^n, v_i \in \mathbb{R}^{n \times k}, \tag{2.9}$$

And $\langle \cdot\, , \cdot \rangle$ is the dot product of two vectors of size k:

$$\langle \mathbf{v}_i, \mathbf{v}_j \rangle = \sum_{f=1}^{k} v_{i,f} v_{j,f} \tag{2.10}$$

A row $v_i$ within $V$ describes the $i^{th}$ variable with k factors.

$k \in \mathbb{N}_0^+$ is a hyperparameter that defines the dimensionality of the factorization.

A 2-way FM (degree d = 2) captures all single and pairwise interactions between variables:

- $w_0$ is the global bias.
- $w_i$ models the strength of the $i^{th}$ variable.
- $\hat{w}_{i,j} := \langle \mathbf{v}_i, \mathbf{v}_j \rangle$ models the interaction between the i-th and j-th variable. Instead of using its own model parameter $\hat{w}_{i,j} \in \mathbb{R}$ for each interaction, the FM models the interaction by factorizing it. We will see later on that this is the key point that allows high-quality parameter estimates of higher-order interactions ($d \geq 2$) under sparsity.

When using the naive method of constructing factorization machine results, the complexity is $O(kn^2)$, but by using a more effective method, it can be made to run in $O(kn)$.

$$\sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j - \frac{1}{2} \sum_{i=1}^{n} \langle \mathbf{v}_i, \mathbf{v}_i \rangle x_i x_i \tag{2.11}$$

$$= \frac{1}{2} \left( \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{f=1}^{k} v_{i,f} v_{j,f} x_i x_j \right) \frac{1}{2} \left( \sum_{i=1}^{n} \sum_{f=1}^{k} v_{i,f} v_{i,f} x_i x_i \right) \tag{2.12}$$

$$= \frac{1}{2} \left( \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{f=1}^{k} v_{i,f} v_{j,f} x_i x_j - \sum_{i=1}^{n} \sum_{f=1}^{k} v_{i,f} v_{i,f} x_i x_i \right) \tag{2.13}$$

$$= \frac{1}{2} \sum_{f=1}^{k} \left( \left( \sum_{i=1}^{n} v_{i,f} x_i \right) \left( \sum_{j=1}^{n} v_{j,f} x_j \right) - \sum_{i=1}^{n} v_{i,f}^2 x_i^2 \right) \tag{2.14}$$

$$= \frac{1}{2} \sum_{f=1}^{k} \left( \left( \sum_{i}^{n} v_{i,f} x_i \right)^2 - \sum_{i=1}^{n} v_{i,f}^2 x_i^2 \right) \tag{2.15}$$

Notice the summing over the possible combinations is the same as summing over the different interactions, minus the self-interactions (divided by two). This is why the value 1/2 is the root of the infinite sequence.

k and n have linear complexity and its complexity is O(kn), substituting this new equation into the existing factorization machine formula, it becomes the following:

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{n} w_i x_i + \frac{1}{2} \sum_{f=1}^{k} \left( \left( \sum_{i}^{n} v_{i,f} x_i \right)^2 - \sum_{i=1}^{n} v_{i,f}^2 x_i^2 \right) \tag{2.16}$$

Factorization machines can perform the following prediction tasks:

- Regression: $\hat{y}(\mathbf{x})$ can be used directly by minimizing the mean squared error between the model prediction and target value, e.g. $\frac{1}{N} \sum^{N} \left(y - \hat{y}(\mathbf{x})\right)^2$
- Classification: for a binary classification, we could minimize the log loss, $\ln\left(e^{-y \cdot \hat{y}(\mathbf{x})} + 1\right)$, where $\sigma$ is the sigmoid or logistic function and y ∈ -1, 1.
- Ranking: the vectors x are ordered by the score of $\hat{y}(\mathbf{x})$ and optimization is done over pairs of instance vectors $(x^(a), x^(b)) \in D$ with a pairwise classification loss (e.g.like in [104]).

To train the factorization machine, we can use a gradient descent-based optimization techniques, and the parameters to be learned are $(\mathbf{w_0}, \mathbf{w}, \mathbf{V})$. The derivation of FM (eq. 2.17) is for the classification loss.

$$\frac{\partial}{\partial \theta} \hat{y}(\mathbf{x}) = \begin{cases} 1, & \text{if } \theta \text{ is } w_0 \\ x_i, & \text{if } \theta \text{ is } w_i \\ x_i \sum_{j=1}^{n} v_{j,f} x_j - v_{i,f} x_i^2 & \text{if } \theta \text{ is } v_{i,f} \end{cases} \tag{2.17}$$

Notice that $\sum_{j=1}^{n} v_{j,f} x_j$ does not depend on i, thus it can be computed independently. The last formula above can also be written as $x_i(\sum_{j=1}^{n} v_{j,f} x_j - v_{i,f} x_i)$. The predicted value $\hat{y}(\mathbf{x})$ is replaced with x for making the notation cleaner.

$$\frac{d}{dx}\left[\ln\left(e^{-yx}+1\right)\right] = \frac{1}{e^{-yx}+1}\cdot\frac{d}{dx}\left[e^{-yx}+1\right] \tag{2.18}$$

$$= \frac{\frac{d}{dx}\left[e^{-yx}\right]+\frac{d}{dx}\left[1\right]}{e^{-yx}+1} \tag{2.19}$$

$$= \frac{e^{-yx}\cdot\frac{d}{dx}\left[-yx\right]+0}{e^{-yx}+1} \tag{2.20}$$

$$= \frac{e^{-yx}\cdot -y}{e^{-yx}+1} \tag{2.21}$$

$$= -\frac{ye^{-yx}}{e^{-yx}+1} \tag{2.22}$$

$$= -\frac{y}{e^{yx}+1} \tag{2.23}$$

## 2.8 Transparency in Factorization Machine Model

Factorization Machines (FMs), a general prediction method that can model high-order function interactions effectively, have been used for various prediction problems. Despite many effective implementations of the algorithm, one of the key drawbacks is transparency. Unfortunately, machine learning models are unable to provide justification for their predictions. Hence, while Factorization Machines predictions might be accurate, it might not be clear to the user why a prediction was generated. Existing FMs do not provide interpretable predictions to users. In FMs, the function interactions are modeled by polynomial expansion, which is difficult to explain to users of the application.

$$\hat{y}(x) = \underbrace{w_0}_{\text{Term 1}} + \underbrace{\sum_{i=1}^{n} w_i x_i}_{\text{Term 2}} + \underbrace{\sum_{i=1}^{n}\sum_{j=i+1}^{n}\langle \mathbf{v}_i, \mathbf{v}_j\rangle x_i x_j}_{\text{Term 3}} \tag{2.24}$$

In the model equation 2.24, Term 1 and Term 2 are classical regression terms that are explainable. Term 3 adds opacity $(v_i, v_j)$ thus reducing transparency of the predictions.

In recent years, there are several efforts have been proposed for FMs to increase the transparency:

#### Subspace Encoding Factorization Machines (SEFM)

SEFM can apply nonlinear feature mapping to both individual features and features that are affected by other features. They first used a nonlinear cut to divide the feature space into bins and then grid cells. Each bin (or grid cell) is given a contribution score for prediction. They obtain the new feature mapping by encoding the feature values into a one-hot vector. The nonzero entries in the new feature representation show the contributions a particular bin or grid cell makes to the model's feature space. The final score of a sample is calculated by combining the scores of the bins and grid cells that the sample is in. Because of this, the SEFM model operates like a scoring system and does not require data binning and score addition (FCB) [105]). It naturally gives interpretable predictions [106], which they present as heat maps of the decision boundaries of classifiers.

#### Knowledge-aware Hybrid Factorization Machine (kaHFM)

The kaHFM is an interpretable model for recommendation systems that relies on factorization machines which are updated to include semantic information encoded in the knowledge graph. The algorithm is trained using semantically valid features from an information graph in order to interpret the machine learning model. With the proposed model, a framework is injected to make the most of the knowledge inherent in the objects. The model's effectiveness and accuracy were checked using two real-life recommendation system datasets. [107].

#### Attentional Factorization Machines (AFM)

AFM strengthens FM by teaching the value of feature interactions within an attention network, which improves the representation of an FM model and allows the model to be more explainable. [108]. The proposed model improves FM by discriminating the importance of feature interactions, which utilizes the recent advance in neural network modeling — the attention mechanism [109] [110] —to enable feature interactions contribute differently to the prediction.

### 2.9 Summary

In this chapter, we reviewed the fundamentals of explainability in machine learning. The next chapter presents a new methodology to make black-box machine learning models more explainable.

# CHAPTER 3

# A NEW EXPLAINABLE MACHINE LEARNING FRAMEWORK BASED ON PRE-HOC AND CO-HOC EXPLAINABILITY

## 3.1 Introduction

We propose two novel approaches to enhancing the explainability of black-box models, which we call *pre-hoc explainability* and *co-hoc explainability*. Our approach aims to incorporate explanations derived from an inherently interpretable white-box model into the original model's learning stage without compromising its high prediction accuracy. Unlike post-hoc explanations, our approach does not rely on input perturbation or secondary model learning, and thus avoids the potential pitfalls of surrogate modeling. Instead, we leverage the insights provided by a white-box model to guide the training of the black-box model in a way that preserves its accuracy while enhancing its global interpretability. We show that our approach outperforms traditional black-box and white-box models on several benchmark datasets and offers a promising direction for making machine learning models more transparent and trustworthy.

## 3.2 Problem Statement

Let $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{\mathcal{N}} \subset \mathcal{Z}$ be a sample from a distribution $\mathcal{D}$ in a domain $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the instance and $\mathcal{Y}$ is the label set. We learn a differentiable predictive function $f \in \mathcal{F} : \mathcal{X} \to \mathcal{Y}$ together with an explainer function $g \in \mathcal{G} : \mathcal{X} \to \mathcal{Y}$ defined over a functional class $\mathcal{G}$. We refer to functions $f$ and $g$ as the predictor and the explainer, respectively, throughout the chapter. $\mathcal{G}$ is strictly constrained to be an inherently explainable functional set, such as a set of linear functions or decision trees. We assume that we have a distance function $d : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ such that $d(y, \hat{y}) = 0 \longleftrightarrow y = \hat{y}$, which measures the point-wise similarity between two probability distributions in $\mathcal{Y}$ and can be used to optimize $f$ and $g$.

Our idea is, instead of learning a post hoc white-box model, to learn a model that is explainable from the start and then let this explainer model guide the predictor model.

To accomplish this goal, we design two different frameworks; (1) A Pre-Hoc Explainable Predictive Framework, where the white box model regularizes the black box model for optimized fidelity and (2) A Co-hoc Explainable Predictive Framework, where white-box and black-box models are optimized simultaneously with a shared loss function that enforces fidelity. See Figure 3.1.

We use the explainer function $g \in \mathcal{G}$ to guide the predictor $f$ by means of distance measures globally. We define global interpretability by measuring how close $f$ is to a family $\mathcal{G}$ over $N$ number of batches in point-wise fashion, see Figure 3.1.

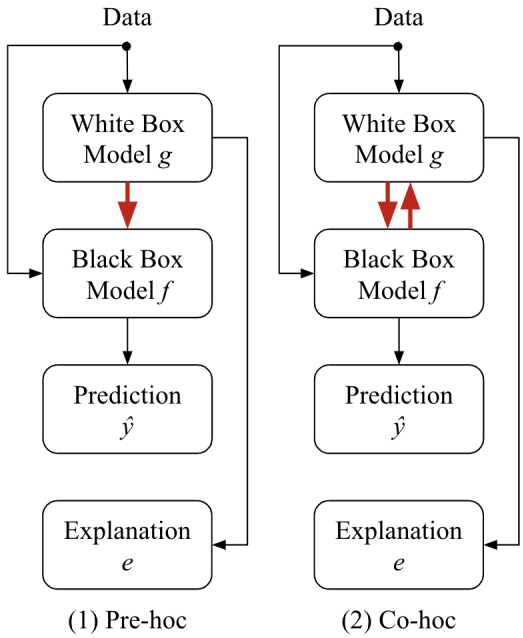



Figure 3.1: Proposed Explainability Frameworks

### 3.3 Enforcing Fidelity

**Definition 3.3.1** (Fidelity Objective Function)**.** *Given an inherently interpretable white-box model $g$ with parameters $\phi$, let its predictions result in a probability distribution $p^{\phi}$. Given the black-box, $f$ with parameters $\theta$, let its predictions result in probability distribution $p^{\theta}$ over $K$ classes $y \in \mathcal{Y} = \{1, 2, .., K\}$. We propose a fidelity objective function, which measures the point-wise probability distance between $p^{\phi}$ and $p^{\theta}$, which are respectively the outputs of $g$ and $f$ for all given input data $\mathcal{X}$. The optimization problem is formulated as:*

$$\min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} D\left(f\left(x_i\right), g\left(x_i\right)\right), \tag{3.1}$$

*where function $D$ is a divergence distance measurement, such as the Jensen-Shannon divergence [111]. We aim to use $D_{JS}$, Jensen-Shannon divergence, to measure the point-wise deviation of the predictive distributions $f_\theta$ and $g_\phi$.*

Denote by $\mathcal{P}$ the set of probability distributions. The Kullback-Leibler divergence (KL). $\mathrm{KL} : \mathcal{P} \times \mathcal{P} \to [0, \infty]$ is a fundamental distance between probability distributions in $\mathcal{D}$ [112], defined by:

$$D_{\mathrm{KL}}(p||q) := \int p \log \frac{p}{q} \, \mathrm{d}\mu, \tag{3.2}$$

where p and q denote probability measures P and Q with respect to $\mu$.

Let $p, q \in \Delta^{K-1}$ have the corresponding weights $\boldsymbol{\pi} = [\pi_1, \pi_2]^T \in \Delta$. Then, the Jensen-Shannon divergence between $p$ and $q$ is given by

$$\begin{aligned} D_{\mathrm{JS}}\left(p, q\right) &:= H(m) - \pi_1 H\left(p\right) - \pi_2 H\left(q\right) \\ &= \pi_1 D_{\mathrm{KL}}\left(p \| m\right) + \pi_2 D_{\mathrm{KL}}\left(q \| m\right), \end{aligned} \tag{3.3}$$

with $H$ the Shannon entropy, and $\boldsymbol{m} = \pi_1 p + \pi_2 q$. Unlike the Kullback-Leibler divergence ($D_{\mathrm{KL}}\left(p \| q\right)$), JS is symmetric, bounded, and does not require absolute continuity.

We propose a fidelity objective function, $L_{JSD}$, that is calculated using the Jensen-Shannon divergence (JS), as follows:

$$\mathcal{L}_{\mathrm{JS}}\left(x_{1:N}, f_\theta, g_\phi\right) := D_{\mathrm{JS}}(\hat{y}^\phi, \hat{y}^\theta) \tag{3.4}$$

$$\begin{aligned} \mathcal{L}_{\mathrm{JS}}\left(x_{1:N}, f_\theta, g_\phi\right) := \frac{1}{2}(&D_{\mathrm{KL}}(\hat{y}^\phi \parallel \frac{(\hat{y}^\phi + \hat{y}^\theta)}{2}) \\ &+ D_{\mathrm{KL}}(\hat{y}^\theta \parallel \frac{(\hat{y}^\phi + \hat{y}^\theta)}{2})) \end{aligned} \tag{3.5}$$

Our goal is to learn the black-box predictive model $f_\theta$ to optimize fidelity to an inherently explainable $g_\phi$.

Substituting Eq. 3.2 into $\mathcal{L}_{JS}$ (Eq. 3.5), we obtain:

$$\begin{aligned}\mathcal{L}_{JS}(\hat{y}^{\phi} \parallel \hat{y}^{\theta}) = \frac{1}{2}(&\sum_{i=1}^{N} \ln\left(\frac{\hat{y}^{\phi}}{\hat{y}^{\theta}}\right)\hat{y}^{\phi} \\ &+ \sum_{i=1}^{N} \ln\left(\frac{\hat{y}^{\theta}}{\hat{y}^{\phi}}\right)\hat{y}^{\theta})\end{aligned} \tag{3.6}$$

Our proposed fidelity objective function has three distinct regularization properties that we explain below.

**Bounded Regularizer** The Jensen–Shannon divergence distance is always bounded, i.e.,

$$0 \leq \mathrm{JS}(p:q) \leq \log 2, \tag{3.7}$$

Since the square root of the JS yields a metric distance satisfying the triangular inequality [113]. Thus, lower and upper bounds become

$$0 \leq D_{\mathrm{JS}}(p:q) \leq \sqrt{\log 2}. \tag{3.8}$$

**Symmetry Preserving Regularizer** The Jensen Shannon divergence is symmetric with respect to two input variables if swapping them does not change the distance. For instance, $D_{\mathrm{JS}}$ is symmetric with respect to p and q if and only if $D_{\mathrm{JS}}(p;q) = D_{\mathrm{JS}}(q;p)$ for all values of p and q. JS is symmetry preserving if the corresponding weights $\boldsymbol{\pi} = [\pi_1, \pi_2]$ are selected as $\boldsymbol{\pi} = \left[\frac{1}{2}, \frac{1}{2}\right]$.

**Differentiable Regularizer** Our fidelity loss implements a differentiable regularizer to enforce fidelity between the predictor model and the explainer model, which is used to derive explanations for the predictor model. The regularizer is based on the Jensen-Shannon divergence (JS) between the probability distributions of the explainer model and the predictor model outputs.

Thus, the regularizer is differentiable, which means that it can be easily incorporated into the training process of the predictor model using standard gradient descent update rules and backpropagation techniques. By minimizing the JS between the two distributions, the regularizer encourages the predictor to produce similar probability distributions to the

explainer model, thereby ensuring that the explanations derived from the explainer model are more accurate and trustworthy. We now present two frameworks for training explainable models to optimize accuracy and fidelity.

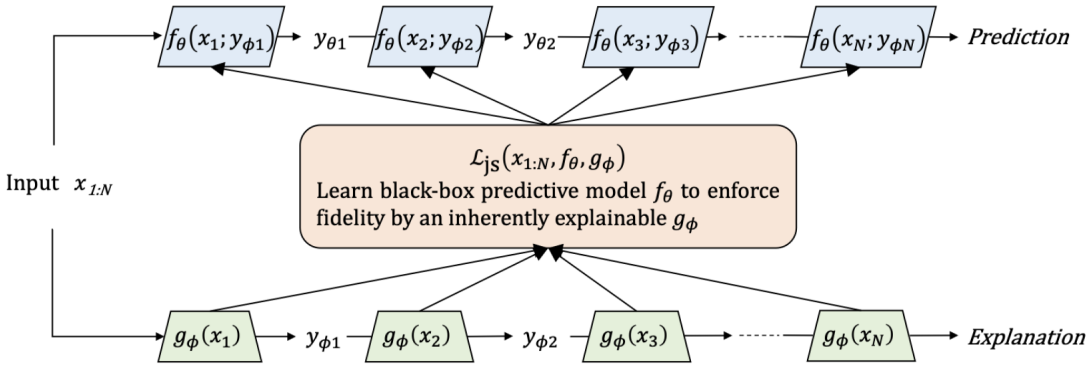



Figure 3.2: Training Phase of Pre-hoc Explainability Framework

### 3.4 Pre-hoc Explainability Framework

We formulate the pre-hoc framework, illustrated in Figure 3.2 (a) and the Algorithm in 3.1, to use a modified learning objective to obtain the Pre-hoc explainability loss as follows

$$\mathcal{L}_{Pre-hoc} = \mathcal{L}_{BCE} + \lambda_1 D_{\text{JS}} + \lambda_2 \mathcal{L}_2, \tag{3.9}$$

where $\mathcal{L}_{BCE}$ is the binary cross-entropy loss that ensure accurate predictions, $\lambda_1$ is an explainability regularization coefficient that controls the smoothness of the new representation and the trade-off between explainability and accuracy, while $\lambda_2$ is the coefficient for standard $\mathcal{L}_2$ regularization of model parameters $\theta$ that aims to avoid overfitting and exploding gradients. Given input data $\mathcal{X}$ and true outputs $y$, explained by the explainer $g$ with parameters $\phi$.

$$\begin{aligned}\mathcal{L}_{Pre-hoc}(\theta, \phi, X, y, ) = & \underbrace{\frac{1}{N}\sum_{i=1}^{N} -y_\theta \log(\hat{y}_\theta) + (1-y_\theta)\log(1-\hat{y}_\theta)}_{\text{Predictor Accuracy}} \\ & + \underbrace{\lambda_1 \frac{1}{2}(\sum_{i=1}^{N} \ln\left(\frac{\hat{y}_\phi}{\hat{y}_\theta}\right)\hat{y}_\phi + \sum_{i=1}^{N} \ln\left(\frac{\hat{y}_\theta}{\hat{y}_\phi}\right)\hat{y}_\theta)}_{\text{Fidelity}} + \underbrace{\lambda_2 \sum_i \theta_i^2}_{\text{L2 Regularization}} \quad ,\end{aligned} \tag{3.10}$$

Hence $\mathcal{L}$ consists of the cross-entropy loss and a fidelity regularization term along with an $\mathcal{L}_2$ regularization term.

Since the explanation $e$ is provided by the white box model $g_\phi$ that is inherently interpretable, the transparency is considered high when the explanatory white box model outputs $\hat{y}_\phi$ are similar to the regularized model $f_\theta$ outputs $\hat{y}_\theta$. This is captured by $D_{\text{JS}}$, which is term 2, Fidelity, in the proposed objective function, $\mathcal{L}_{\text{Pre-hoc}}$ (eq. 3.9). While the objective function is to learn the model that will make accurate predictions, we give greater importance to the model's predictions that are similar to the white-box predictions and penalize those that are not similar.

**Algorithm 3.1** Pre-hoc Explainability Framework

**Require:** Black-box model $f_\theta$, white-box model $g_\phi$, input instance $x$, true label $y$, and parameter $\lambda_1$ for weighting the divergence term.

**procedure** PREHOCEXPLAINABILITY($f_\theta$, $g_\phi$, $x$, $y$, $\lambda_1$)

**for** each $x_i$ in $X_{train}$ **do**

Compute $p^\phi = g_\phi(x_i)$ ▷ Predictions from white-box model

Compute $p^\theta = f_\theta(x_i)$ ▷ Predictions from black-box model

Compute $L_{GJS} = \text{GJS}(p^\theta, \boldsymbol{p}^\phi)$ ▷ Using GJS divergence

Compute $L_{BCE} = \text{BinaryCrossEntropy}(p^\theta, \text{y})$

$L_{total} = L_{BCE} + \lambda_1 \cdot L_{GJS}$

Update $f_\theta$ using gradient descent: $\theta \leftarrow \theta - \alpha \nabla_\theta L_{total}$

**end for**

**end procedure**

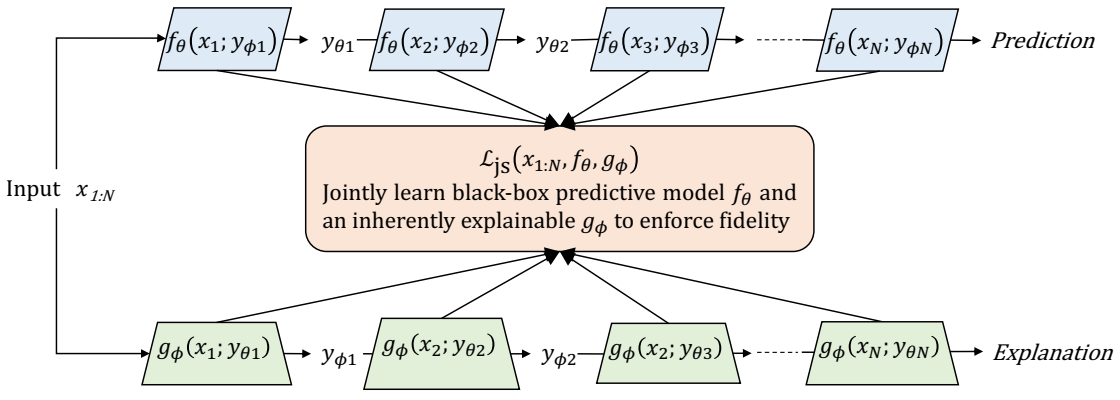



Figure 3.3: Training Phase of Co-hoc Explainability Framework

### 3.5 Co-hoc Explainability Framework

We formulate the Co-hoc explainability framework illustrated in Figure 3.3 and the Algorithm in 3.2, to use a modified learning objective to obtain the Co-hoc explainability loss as follows

**Definition 3.5.1** (Co-hoc Fidelity Objective Function)**.** *Given an inherently interpretable white-box model $g$ with parameters $\phi$, let its predictions result in a probability distribution $p^{\phi}$ and given the black-box model $f$ with parameters $\theta$, let its predictions result in probability distribution $p^{\theta}$ over $K$ classes $y \in \mathcal{Y} = \{1, 2, .., K\}$. We propose a Co-Learning Explainability Framework, where $f_\theta$ and $g_\phi$ are jointly learned, given $p^{\phi}$ and $p^{\theta}$, respectively, as inputs. We use an added distance function (eq. 4.1) as a regularization for the objective function to guide the co-learning process. Our global distance metric is the same as the Definition 4.3.1, and the combined Co-hoc loss function is given by*

$$
\begin{aligned}
\mathcal{L}_{Co-hoc}(\theta, \phi, X, y, ) = & \underbrace{\frac{1}{N}\sum_{i=1}^{N} -y_n \log\left(\hat{y}_{\theta n}\right) + (1-y_n)\log\left(1-y_{\hat{\theta} n}\right)}_{Predictor\ Accuracy} \\
& + \underbrace{\frac{1}{N}\sum_{i=1}^{N} -y_n \log\left(\hat{y}_{\phi n}\right) + (1-y_n)\log\left(1-y_{\hat{\phi} n}\right)}_{Explainer\ Accuracy} \\
& + \underbrace{\lambda_1 \frac{1}{2}\left(\sum_{i=1}^{N} \ln\left(\frac{\hat{y}_{\phi}}{y_{\hat{\theta} n}}\right)\hat{y}_{\phi n} + \sum_{i=1}^{N} \ln\left(\frac{y_{\hat{\theta} n}}{\hat{y}_n}\right) y_{\hat{\theta} n}\right)}_{Fidelity} \\
& + \underbrace{\lambda_2 \sum_i \theta_i^2}_{Regularization\ 1} + \underbrace{\lambda_3 \sum_i \phi_i^2}_{Regularization\ 2} ,
\end{aligned}
\tag{3.11}
$$

*which contains binary cross-entropy and fidelity regularization terms, along with other regularization terms; Regularization 1 discourages exploding gradients, and Regularization 2 encourages the sparsity of the explainer model.*

The primary distinction between Co-hoc and Pre-hoc lies in the *joint* optimization of predictor $f_\theta$ and explainer $g_\phi$ through simultaneous stochastic gradient descent with mini-batches, see Figure 3.3.

The gradient can be calculated as:

$$\frac{d}{d\hat{y}^{WB}}\left[\frac{1}{2}(\sum_{i=1}^{N}\ln\left(\frac{\hat{y}^{WB}}{\hat{y}^{BB}}\right)\hat{y}^{WB}+\sum_{i=1}^{N}\ln\left(\frac{\hat{y}^{BB}}{\hat{y}^{WB}}\right)\hat{y}^{BB})\right] \tag{3.12}$$

$$=\frac{1}{2}N\left[\frac{d}{d\hat{y}^{WB}}(\hat{y}^{WB}\log\left(\frac{\hat{y}^{WB}}{\hat{y}^{BB}}\right))+\frac{d}{d\hat{y}^{WB}}(\hat{y}^{BB}\log\left(\frac{\hat{y}^{BB}}{\hat{y}^{WB}}\right))\right] \tag{3.13}$$

$$=\frac{1}{2}N\left[\left[\log\left(\frac{\hat{y}^{WB}}{\hat{y}^{BB}}\right)(\frac{d}{d\hat{y}^{WB}}(\hat{y}^{WB}))+\hat{y}^{WB}\frac{d}{d\hat{y}^{WB}}(\log\left(\frac{\hat{y}^{WB}}{\hat{y}^{BB}}\right))\right]+\hat{y}^{BB}\frac{d}{d\hat{y}^{WB}}(\log\left(\frac{\hat{y}^{BB}}{\hat{y}^{WB}}\right))\right] \tag{3.14}$$

$$=\frac{1}{2}N\left[\left[\log\left(\frac{\hat{y}^{WB}}{\hat{y}^{BB}}\right)(1)+\hat{y}^{WB}(\frac{1}{\hat{y}^{WB}})\right]+\hat{y}^{BB}(-\frac{1}{\hat{y}^{WB}}))\right] \tag{3.15}$$

$$=\frac{1}{2}N\left[\left[\log\left(\frac{\hat{y}^{WB}}{\hat{y}^{BB}}\right)+(1)\right]-(\frac{\hat{y}^{BB}}{\hat{y}^{WB}})\right] \tag{3.16}$$

$$=\frac{1}{2}N\left[\log\left(\frac{\hat{y}^{WB}}{\hat{y}^{BB}}\right)-\frac{\hat{y}^{BB}}{\hat{y}^{WB}}+1\right] \tag{3.17}$$

**Algorithm 3.2** Co-hoc Explainability Framework

**Require:** Black-box model $f_\theta$, white-box model $g_\phi$, input instance $x$, true label $y$, and parameter $\lambda_1$ for weighting the divergence term.

**procedure** CoHocExplainability($f_\theta$, $g_\phi$, $x$, $y$, $\lambda_1$)

**for** each $x_i$ in $X_{train}$ **do**

Compute $p^\phi = g_\phi(x_i)$ ▷ Predictions from white-box model

Compute $p^\theta = f_\theta(x_i)$ ▷ Predictions from black-box model

Compute $L_{GJS} = \text{GJS}(p^\theta, p^\phi)$ ▷ Using GJS divergence

Compute $L_{BCE_\theta} = \text{BinaryCrossEntropy}(p^\theta, \text{y})$

Compute $L_{BCE_\phi} = \text{BinaryCrossEntropy}(p^\phi, \text{y})$

$L_{total} = L_{BCE_\theta} + L_{BCE_\phi} + \lambda_1 \cdot L_{GJS}$

Update $f_\theta$ using gradient descent: $\theta \leftarrow \theta - \alpha\nabla_\theta L_{total}$

Update $g_\phi$ using gradient descent: $\phi \leftarrow \phi - \alpha\nabla_\phi L_{total}$

**end for**

**end procedure**

### 3.6 Generating Explanations

In our proposed explainable machine learning frameworks, we generate global explanations using a sparse linear model, considered an inherently explainable white-box model. The sparse linear model allows us to provide interpretable feature importance scores, enabling users to understand the contribution of each feature to the model's predictions. The feature importance scores provide a quantitative measure of the relative importance of each feature in the model's decision-making process. Algorithm 3.3 outlines the process of cal-

culating feature importance scores based on the trained white-box model and the input dataset.

The algorithm for calculating the importance score of the features takes two inputs: the trained white-box model $g$ and the dataset $Z$ containing the features. The output of the algorithm is a vector of feature importance scores, where each score corresponds to a specific feature in the dataset.

**Algorithm 3.3** Feature Importance Score Calculation

**Require:** Trained explainer model $g$, dataset with features $Z$

1: $featureImportances \leftarrow \{\}$ ▷ Initialized as an empty list
2: Calculate the mean absolute deviation $MAD_A$ for each attribute $A$ in the feature set set $Z$ using Eq. 3.18
3: **for** each attribute $A$ in the feature set $Z$ **do**
4: Extract the coefficient parameters or weights $\phi_A$ associated with feature $A$ from model $g$
5: Compute importance-score $= \phi_A * MAD_A$
6: Append importance-score to $featureImportances$
7: **end for**
8: $TopKfeatureImportances = TopK(Sort_{descending}(featureImportances))$
9: **return** $TopKfeatureImportances$ ▷ return list of feature importance scores

The algorithm calculates the mean absolute deviation (MAD) for each feature in the dataset $Z$. The MAD is a robust measure of the distribution that quantifies the average absolute deviation of each data point from the mean. The MAD of Attribute $A$ is calculated as follows:

$$MAD_A = \frac{1}{N} \sum_{i=1}^{N} |A_i - \bar{A}| \tag{3.18}$$

where $N$ is the number of data points, $A_i$ is Attribute $A$ of the $i$-th data point, and $\bar{A}$ is the mean of attribute $A$ in the dataset.

MAD is used as a scaling factor to normalize feature importance scores, ensuring that the scores are comparable across different features and datasets. By incorporating the MAD in the importance score calculation, we account for the variability and scale of the features, providing a more reliable and interpretable measure of feature importance. After calculating the MAD, the algorithm iterates over each feature $g$ in the model's parameters $\theta$ and extracts the corresponding weights from the trained white-box model for each feature. These weights represent the learned coefficients that determine the impact of each feature on the model's predictions. To compute the importance score for a specific feature, the

algorithm multiplies the mean weight of the feature by its corresponding MAD value. The mean weight is obtained by averaging the weights associated with the feature across all instances in the dataset. This step ensures that the importance score considers the feature's overall contribution to the model's predictions.

## 3.7 Experimental Results

In this section, we first give a brief description of the data and metrics used to evaluate our model. Then, we present the results and analyze the experimental results for the methods proposed in Section 3.

### 3.7.1 Research Questions

We conduct experiments aimed at answering the following research questions:

**RQ1:** Can we maintain an accuracy that is higher than the explainer model even if it is lower than the baseline BB predictor model; while having explanations from $g_\phi$ (since $g_\phi$ was used to guide $f_\theta$?)

**RQ2:** How good is our regularized predictor model $f_\theta$ at mimicking the explainer model $g_\phi$?

**RQ3:** How does $\lambda_1$ affect the fidelity and accuracy trade-off?

**RQ4:** What are the differences between the pre-hoc and co-hoc frameworks?

### 3.7.2 Datasets

We experimented with three publicly accessible real-world datasets. All three datasets used in a binary classification setting. Movielens 100k movie ratings, has 100,000 ratings based on 1000 users on 1700 movies. MovieLens 1M movie ratings, has 1 million ratings based on 6000 users on 4000 movies. For Movielens datasets [114], the classification target is the movie rating. Our goal is to learn like or dislike a movie. The target is discretized into liked and disliked; 1 is the class label for a rating of 3 and above; 0 is the class label for a rating of less than 3. FICO HELOC dataset [115] contains 10,459 anonymized information about home equity line of credit (HELOC) applications made by real homeowners. The target variable is risk performance, which predicts whether the homeowner qualifies for a line of credit or not.

### 3.7.3 Evaluation Protocols

To assess the classification accuracy, we use the Area under the ROC Curve, AUC($f_\theta$, $\hat{y}$). Each dataset is split randomly into training, validation, and test sets in the ratio 80:10:10. After training on every batches with a learning rate of 0.001, AUC is calculated on the validation and test datasets. We measure all the metrics on a held-out test set. All models are trained with $L_2$ regularization until validation accuracy is stabilized for at least ten epochs.

Fidelity, also known as descriptive accuracy [116], measures how accurately an explanation method can mimic the behavior of a black-box classifier in terms of assigning class labels to data records. We use AUC($f_\theta$, $g_\phi$) to evaluate the fidelity. Our baseline for fidelity is the AUC of the original black-box predictor model and the explainer model, which can also be considered as a post-hoc explainability score without any optimization.

### 3.7.4 Baselines

We compared our Pre-hoc and Co-hoc predictor models with their original black-box (BB) version. The black-box model is Factorization Machines [117] as it is a widely used state of the art for classification, regression, and recommendation tasks. The explainer white-box model (WB) is a sparse logistic regression model, which is inherently explainable, and thus provides the explanation.

### 3.7.5 Parameter Settings

We implemented our proposed methods based on PyTorch. All models are learned by optimizing the binary cross entropy with Adam [118], which is an extension to stochastic gradient descent. Batch size is selected as 64,2056,64 respectively, for ML-100K, ML-1M, and HELOC datasets, which are the optimal batch size for each dataset. We tested for $\lambda_1$ in $\{0.01, 0.1, 0.25, 0.5, 0.75, 1\}$. The regularization weight of the loss function is estimated using a mini-batch. We pick the best regularization weight for each dataset using the validation set and use that for the final evaluation. The final evaluation is done by retraining the models using their chosen configurations and evaluating them on the test set.

TABLE 3.1: Proposed Explainability Frameworks, $\lambda_1$ comparison in prediction accuracy (AUC) and explainability (Fidelity) on the three datasets that were described in experimental settings. Higher AUC and Fidelity is better.

| Framework | Dataset | Metric | $\lambda_1 = 0.01$ | $\lambda_1 = 0.1$ | $\lambda_1 = 0.25$ | $\lambda_1 = 0.5$ | $\lambda_1 = 0.75$ | $\lambda_1 = 1$ |
|---|---|---|---|---|---|---|---|---|
| Pre-hoc | ml-100k | AUC | 0.7840 | 0.7845 | 0.7849 | **0.7841** | 0.7801 | 0.7740 |
| | | Fidelity | 0.8207 | 0.8283 | 0.8430 | 0.8755 | 0.9094 | **0.9410** |
| | ml-1M | AUC | 0.8075 | 0.8079 | **0.8076** | 0.8033 | 0.7954 | 0.7896 |
| | | Fidelity | 0.8769 | 0.8871 | 0.9058 | 0.9404 | 0.9696 | **0.9856** |
| | HELOC | AUC | 0.7591 | 0.7611 | 0.7648 | 0.7699 | **0.7720** | 0.7719 |
| | | Fidelity | 0.7482 | 0.7541 | 0.7664 | 0.7903 | 0.8137 | **0.8454** |
| Co-hoc | ml-100k | AUC | 0.7840 | 0.7845 | 0.7852 | **0.7849** | 0.7816 | 0.7766 |
| | | Fidelity | 0.8215 | 0.8326 | 0.8507 | 0.8869 | 0.9194 | **0.9492** |
| | ml-1M | AUC | 0.8075 | 0.8079 | **0.8077** | 0.8036 | 0.7968 | 0.7924 |
| | | Fidelity | 0.8771 | 0.8901 | 0.9122 | 0.9484 | 0.9749 | **0.9868** |
| | HELOC | AUC | 0.7591 | 0.7612 | 0.7651 | 0.7707 | 0.7736 | **0.7743** |
| | | Fidelity | 0.7482 | 0.7563 | 0.7705 | 0.7767 | 0.8277 | **0.8572** |

### 3.7.6 Results and Discussion

**RQ1: For the predictor model $f_\theta$, can we maintain an accuracy that is higher than the explainer model $g_\phi$ if lower than the original $f$; while having explanations from $g_\phi$ (since $g_\phi$ was used to guide the $f_\theta$)?**

The experimental results in Table 3.2, Figure 3.4, Figure 3.5 provide insights into this question. Comparing the accuracy scores between the predictor model and the explainer model, we observe that the predictor model for both Pre-hoc predictor and Co-hoc predictor consistently achieves significantly higher accuracy scores than the explainer model $g_\phi$ (p-value $< .05$), even with the $\lambda_1 = 1$, which is the highest coefficient for optimizing fidelity, for each dataset and the both proposed models.

These results clearly demonstrate that the proposed Pre-hoc and Co-hoc predictors maintain higher accuracy compared to the explainer model $g_\phi$ baseline while achieving improved fidelity in mimicking the behavior of the explainer model $g_\phi$.

**RQ2: How good is our regularized predictor model $f_\theta$ in mimicking the explainer model $g_\phi$?**

On the ml-100k dataset, Pre-hoc predictor and Co-hoc predictor achieve fidelity

TABLE 3.2

Model comparison in terms of prediction accuracy and fidelity of explainability on the three real-world datasets, two interaction datasets, ML100k and ML1M, and one tabular, HELOC dataset. All evaluation metrics are computed with, respectively, for datasets $\lambda_1 = 0.75, 0.5, 1$ and batch size $n = 64, 2056, 64$. The explainer is the white-box model, BB is the predictor black-box model. The best results are in **bold**. Higher AUC and Fidelity is better.

| Dataset | ml-100k | | ml-1M | | HELOC | |
|---|---|---|---|---|---|---|
| Model | AUC | Fidelity | AUC | Fidelity | AUC | Fidelity |
| Explainer WB | 0.7655 | - | 0.7882 | - | 0.7616 | - |
| Original BB | 0.7784 | 0.8287 | **0.8078** | 0.8875 | 0.7703 | 0.7728 |
| Pre-hoc BB | 0.7801 | 0.9094 | 0.8033 | 0.9404 | 0.7698 | 0.8454 |
| Co-hoc BB | **0.7816** | **0.9194** | 0.8036 | **0.9484** | **0.7743** | **0.8572** |

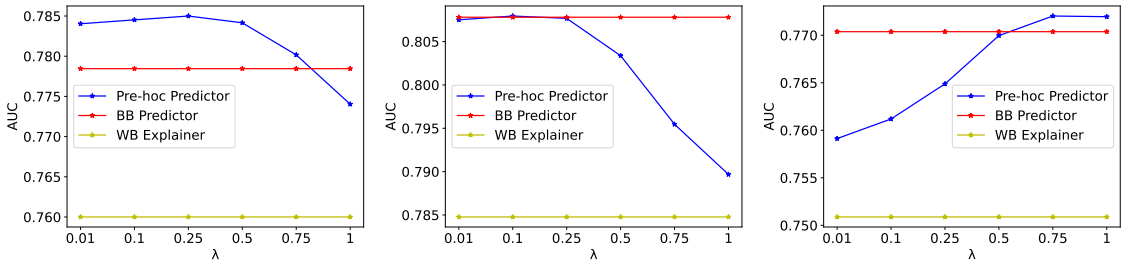
scores of 0.9094 and 0.9194, respectively, 9.7% and 10.9% increase by outperforming the original black-box predictor fidelity score of 0.8287. This indicates that the proposed models better capture the behavior of the explainer model compared to the baseline predictor model. Similarly, on the ml-1M dataset, Pre-hoc predictor and Co-hoc predictor achieve a fidelity score of 0.9404 and 0.9484, outperforming the original fidelity score of 0.8875 by improving 5.9% and 6.8%. Moreover, in the HELOC dataset, improvement in the fidelity score is respectively, 9.3% and 10.9% for Pre-hoc predictor and Co-hoc predictor.

In summary, the experimental findings support the notion that the proposed models, Pre-hoc and Co-hoc predictors, are more successful in mimicking the behavior of the explainer model compared to the baseline black-box model. This demonstrates the effectiveness of the proposed techniques in enhancing the fidelity of the predictor model while maintaining high accuracy.

Importantly, these improvements in fidelity are achieved without any decrease in the accuracy score. The proposed predictor models maintain a similar level of accuracy compared to the original black-box predictor. The trade-off between fidelity and accuracy will be explored and discussed further in the next research question.

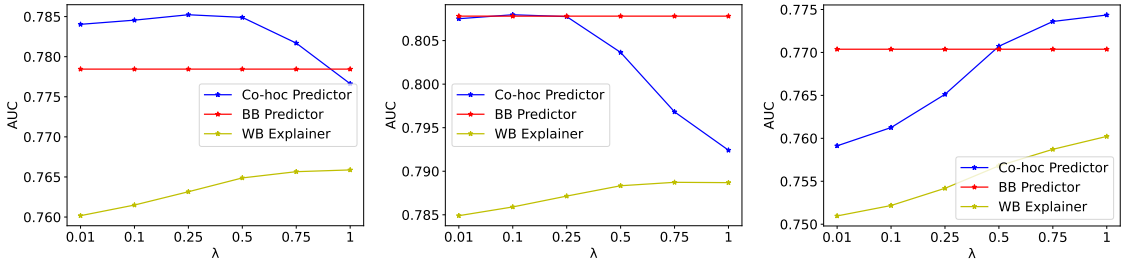
**RQ3: How does lambda affect the fidelity and accuracy trade-off?**

As we can see in Table 3.1, the regularization hyperparameter $\lambda_1$ plays a crucial




(a) (b) (c)

Figure 3.4: Pre-hoc Explainability Framework Comparison in Accuracy AUC for different $\lambda_1$ on the ml-100k (a), ml-1M (b), HELOC (c) datasets. Pre-hoc Predictor is our proposed model, BB is the original black-box predictor model, WB is the explainer model.




(a) (b) (c)

Figure 3.5: Co-hoc Explainability Framework Comparison in Accuracy AUC for different $\lambda_1$ on the ml-100k (a), ml-1M (b), HELOC (c) datasets. Co-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model.

role in balancing the fidelity and accuracy trade-off in our models. A value of $\lambda_1 = 0$ indicates that there is no regularization of the fidelity, while $\lambda_1 = 1$ signifies an equal weight of the fidelity and accuracy in the objective function. The impact of $\lambda$ on the results is noteworthy. When $\lambda_1$ is set to 0.01 and 0.1, there is no noticeable difference in the Fidelity AUC metric across all datasets. However, as we increase the value of $\lambda_1$, we observe a consistent improvement in the fidelity results. In particular, when $\lambda_1$ is set to 1, we achieve almost perfect fidelity scores, indicating a high level of agreement between the explainer model $g_\phi$ and the regularized predictor model $f_\theta$. This demonstrates the effectiveness of the regularization approach in mimicking the behavior of the explainer model.

In the Pre-hoc Explainability Framework, for the ml-100k dataset, the accuracy values remain relatively stable across different $\lambda_1$ values, ranging from 0.7840 to 0.7740. On the other hand, the fidelity values gradually increase as $\lambda_1$ increases, starting from 0.8207 and reaching a peak at $\lambda$=1.0 with a fidelity value of 0.9410, which is a 14.6% increase. This

suggests that higher $\lambda_1$ values in the Pre-hoc framework result in more faithful explanations without significantly sacrificing accuracy.

Similarly, for the ml-1M dataset, the accuracy values are relatively consistent, with the highest accuracy, 0.8076, observed at $\lambda_1$= 0.25. On the other hand, the fidelity values show an increase of at most 12. 3% when the value of $\lambda_1$ increases, ranging from 0.8769 to 0.9856. $\lambda_1$=1.0 achieves the highest fidelity, indicating that the Pre-hoc framework with a higher $\lambda_1$ value captures more accurate and informative explanations.

In the HELOC dataset, the accuracy values range from 0.7591 to 0.7719 across different $\lambda_1$ values. As $\lambda_1$ increases, the fidelity values also exhibit an upward trend, starting from 0.7482 and reaching a peak at $\lambda$=1.0 with a fidelity value of 0.8454. This suggests that the Pre-hoc framework with higher $\lambda$ values enhances the fidelity of the explanations while maintaining comparable accuracy.

Similarly, in the Co-hoc Explainability Framework; for the ml-100k dataset, the accuracy values remain relatively stable, ranging from 0.7840 to 0.7766. The fidelity values show a gradual increase with increasing $\lambda_1$, starting from 0.8215 and reaching the highest fidelity of 0.9492 at $\lambda_1$=1. This suggests that the Co-hoc framework with a moderate $\lambda$ value achieves better fidelity without compromising accuracy significantly.

In the ml-1M dataset, the accuracy values are consistent across different $\lambda_1$ values, ranging from 0.8075 to 0.7924. On the other hand, the fidelity values show an increase of 12. 5%, starting at 0.8771 and reaching the highest fidelity of 0.9868 at $\lambda_1$=1. This indicates that the Co-hoc framework with higher $\lambda$ values captures more faithful explanations while maintaining comparable accuracy.

In the HELOC dataset, the accuracy values range from 0.7591 to 0.7743 across different $\lambda_1$ values. As $\lambda_1$ increases, the fidelity values also exhibit an upward trend with a 14.5% increase, starting from 0.7482 and reaching the highest fidelity of 0.8572 at $\lambda_1$=1. This suggests that the Co-hoc framework with higher $\lambda_1$ values improves the fidelity of the explanations while maintaining similar accuracy levels.

Overall, the analysis of the experimental results shows that increasing $\lambda_1$ values lead to improvements in fidelity, indicating more accurate and faithful explanations with accuracy values remaining relatively stable or showing marginal variations across different $\lambda_1$ values. The optimal balance between accuracy and fidelity may vary depending on the dataset and the specific requirements of the explainability framework.

**RQ4: What are the differences between the pre-hoc and co-hoc frameworks?**

For comparing the two proposed frameworks, we consider accuracy, fidelity, and lambda sensitivity properties aspects.

**Accuracy**: In terms of prediction accuracy, both frameworks generally perform similarly across the evaluated datasets. In most cases, the accuracy values are comparable, with only slight variations observed. For example, in the ml-100k dataset, both frameworks achieve accuracy values around 0.78. Similarly, in the ml-1M and HELOC datasets, both frameworks achieve similar accuracy values with slight differences.

**Fidelity**: The fidelity of the Co-hoc framework tends to be consistently higher compared to the Pre-hoc framework. Across all datasets, the Co-hoc framework achieves higher fidelity scores, indicating that it better approximates the behavior of the explainer model. For instance, in the ml-100k dataset, the fidelity of Co-hoc ranges from 0.8215 to 0.9492, whereas Pre-hoc ranges from 0.8207 to 0.9410. The same trend applies to other datasets as well, see Table 3.1.

**Sensitivity to Regularization Coefficient** $\lambda$: Both frameworks exhibit sensitivity to the choice of $\lambda_1$. The performance in terms of accuracy and fidelity can vary depending on the specific value of $\lambda_1$ used. The optimal value of $\lambda_1$ that maximizes the trade-off between fidelity and accuracy may differ between the two frameworks and across different datasets.

Overall, the Co-hoc Explainability Framework consistently demonstrates higher fidelity than the Pre-hoc Framework, while the differences in prediction accuracy between the two frameworks are relatively minor. This suggests that the Co-hoc approach, which jointly optimizes both the predictor model and explainer model, has the potential to approximate the mechanisms of the original model better and provide more accurate explanations. Another advantage of the co-hoc framework over the pre-hoc framework is that it learns a more accurate white-box expaliner model due to the joint training phase.

## 3.8 Summary

We proposed two novel explainability frameworks based on pre-hoc and co-hoc explainability. The Pre-hoc explainability and Co-hoc explainability frameworks guide the black-box predictor model's training with an interpretable white-box model to align the black-box predictor's global logic with the white-box explainer's transparent reasoning rather than extracting post-hoc approximations of the white-box's logic. The proposed models incorporate the fidelity for any differentiable machine learning model without modifying the model architecture. Our work addresses the lack of explainability optimization during training and model-agnostic methods to enhance global explainability.

Our experimental results evaluated the methods in comparison with the WB and BB LR and FM baselines, respectively, showing improvements in terms of accuracy, in addition to providing an explanation that contributes to making the predictions explainable.

One limitation of the proposed methods in this chapter is that they only provide a single global explanation model for the entire data set. In the the next chapter, we propose new algorithms to extend our framework to produce *local* explanations that can better explain the predictions for different instances of the data set.

# CHAPTER 4

# OPTIMIZING IN-TRAINING EXPLAINABLE MACHINE LEARNING FOR LOCAL EXPLAINABILITY

## 4.1 Introduction

One limitation of the methods proposed in the previous chapter is that they only provide a single *global* explanation model for the entire data set. While *global* explanations, as we have shown in the previous chapter, can provide an overall interpretation of a model's behavior, *local* explanations offer insights into the model's decision-making process for *individual* instances. Local explainability is thus particularly important in domains such as healthcare, finance, and criminal justice, where the different consequences of individual predictions can be significant, and knowing the factors that affect a *specific* decision is crucial.

In this chapter, we extend our in-training explainable machine learning framework to incorporate local explainability, enabling the generation of instance-specific explanations. We present the problem formulation, introduce the concept of local explainability, and propose an algorithm to enforce *local* fidelity by leveraging the Jensen-Shannon divergence between the predictor and the explainer models using neighborhood information.

## 4.2 Problem Statement

Let $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{\mathcal{N}} \subset \mathcal{Z}$ be a sample from a distribution $\mathcal{D}$ in a domain $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is the instance space and $\mathcal{Y}$ is the label space. We learn a differentiable *predictive* function $f \in \mathcal{F} : \mathcal{X} \to \mathcal{Y}$ together with a transparent *explainer* function $g \in \mathcal{G} : \mathcal{X} \to \mathcal{Y}$ defined over a functional class $\mathcal{G}$. We refer to functions $f$, $g^{global}$, and $g^{local}$ as the *predictor*, *global explainer* and *local explainer*, respectively. $\mathcal{G}$ is strictly constrained to be an inherently explainable functional set, such as a set of linear functions.

Our goal is to optimize the predictor $f$ to provide global explanations that are faithful to the model's behavior and consistent with the explanations generated by the global

TABLE 4.1

Notation

| Symbol | Description |
|---|---|
| $f_\theta$ | Predictor: black-box machine learning model with parameters $\theta$ |
| $g_\phi^{global}$ | Global Explainer: white-box machine learning model with parameters $\phi$ |
| $g_{\phi_i}^{local}$ | Local white-box Explainer model for testing instance $x_i$, with parameters $\phi_i$ |
| $p_\phi$ | Probability distribution of the outputs of $f_\theta$ |
| $p_\theta$ | Probability distribution of the outputs of $g_\phi$ |
| $L_2$ | Regularization for sparsity |
| $D$ | Divergence measurement |
| $JS$ | Jensen-Shannon divergence |
| $\lambda$ | Explainability regularization coefficient |
| $BB$ | Black-box machine learning model |
| $WB$ | White-box machine learning model |
| $x$ | Input instance |
| $y$ | True label or target |
| $\hat{y}$ | Predicted target |
| $\hat{y}_\theta$ | Predicted target label from the predictor |
| $\hat{y}_\phi$ | Predicted target label from the explainer |
| $\mathcal{N}_{x_i}^{training}$ | Local in-training neighborhood instances of data point $x_i$ |
| $\mathcal{N}_{x_i}^{testing}$ | Local in-testing neighborhood instances of data point $x_i$ |
| $getNeighbors$ | Neighborhood function |
| $Z$ | Dataset |
| $PF$ | Point Fidelity |
| $NF$ | Neighborhood Fidelity |
| $TV$ | Total Variation |

explainer $g^{global}$. Then, fine-tune the global explainer $g^{global}$ within the local neighborhood, $\mathcal{N}(x_i^{testing})$, of a new testing instance $x_i$ to obtain local explainer $g_i^{local}$. We aim to achieve this by minimizing the divergence between the outputs of the predictor and global explainer in the local neighborhood while maintaining the predictor's accuracy and generating local explanations from the local explainer model, $g_i^{local}$.

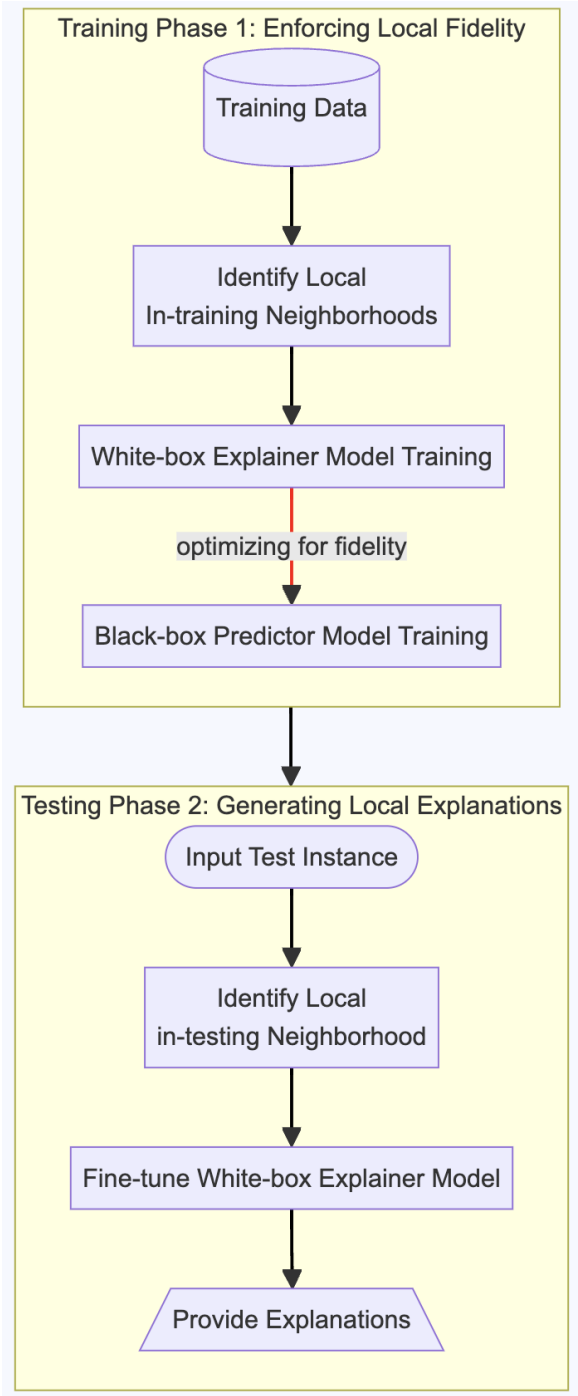



Figure 4.1: Pre-hoc Local Explainable Machine Learning Framework consists of two phases. Phase 1: Training for Fidelity is the training phase that optimizes the agreement between the white-box explainer and black-box predictor models, quantified by fidelity. Phase 2: Generating a local explanation for a new test instance by fine-tuning the white-box explainer model within its neighboring training instances.

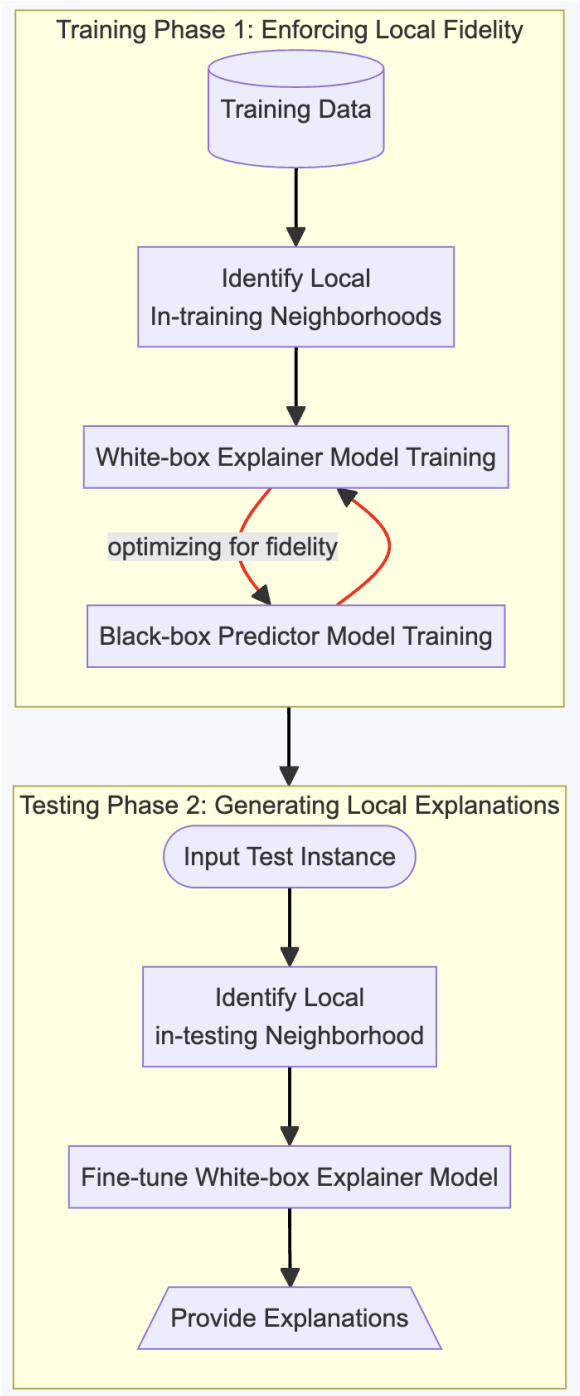



Figure 4.2: Co-hoc Local Explainable Machine Learning Framework consists of 2 phases. Phase 1: Training for Fidelity is the training phase of the framework that co-optimizes the agreement between the white-box explainer and black-box predictor models, quantified by fidelity. Phase 2: Generating a local explanation for a new test instance by fine-tuning the white-box explainer model within its neighboring training instances.

## 4.3 Optimizing Local Explainability Using the Neighbors of an Instance

Local explainability refers to understanding and interpreting a model's predictions at an individual instance level, whereas global explanations provide an overall understanding of the model's behavior. Thus, local explanations can offer insights into the factors influencing a specific prediction. In the following, we present the definitions that will be used to build the proposed methodology.

**Definition 4.3.1** (Neighborhood Fidelity Objective Function)**.** *Given a global explainer model $g_\phi^{global}$ with parameters $\phi$, let its predictions result in a probability distribution $p_\phi$. Given the predictor, $f_\theta$ with parameters $\theta$, let its predictions result in probability distribution $p_\theta$ over $K$ classes $y \in \mathcal{Y} = \{1, 2, .., K\}$. We propose a neighborhood fidelity objective function, which measures the probability distances of local in-training neighborhoods $\mathcal{N}^{training}(x_i)$ of instance $i$, between $p_\phi$ and $p_\theta$, which are respectively the outputs of $g_\phi^{global}$ and $f_\theta$ for all given input training data $\mathcal{X}$. The optimization problem is formulated as follows:*

$$\min_{f_\theta \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} D_{local} \left( f_\theta \left( \mathcal{N}^{training}(x_i) \right), g_\phi^{global} \left( \mathcal{N}^{training}(x_i) \right) \right), \tag{4.1}$$

*Where function $D_{local}$ is a divergence distance measurement, such as the Jensen-Shannon divergence [111], $D_{JS_{local}}$, to measure the within-neighborhood deviation between the predictive distributions of $f_\theta$ and $g_\phi^{global}$.*

Denote the set of probability distributions by $\mathcal{P}$. The Kullback-Leibler divergence (KL) $\mathrm{KL} : \mathcal{P} \times \mathcal{P} \to [0, \infty]$ is a fundamental distance between probability distributions in $\mathcal{D}$ [112], defined on a neighborhood $_\mathcal{N}$ by:

$$D_{\mathrm{KL},\mathcal{N}}(p||q) := \int_{\mathcal{N}} p \log \frac{p}{q} \, \mathrm{d}\mu, \tag{4.2}$$

where $p$ and $q$ denote probability measures P and Q with respect to $\mu$.

Let $p, q \in \Delta^{K-1}$ have the corresponding weights $\boldsymbol{\pi} = [\pi_1, \pi_2]^T \in \Delta$. Then, the Jensen-Shannon divergence between $p$ and $q$ is given by

$$D_{\mathrm{KL},\mathcal{N}}(p, q) := \pi_1 D_{\mathrm{KL},\mathcal{N}}(p \| m) + \pi_2 D_{\mathrm{KL},\mathcal{N}}(q \| m), \tag{4.3}$$

Finally, $D_{\mathrm{JS}_{\mathrm{local}}}$ is obtained as follows:

$$D_{\mathrm{JS}_{\mathrm{local}}}(\hat{y}_\phi, \hat{y}_\theta) = \frac{1}{2}\Big(\sum_{x_j \in \mathcal{N}_{x_i}^{training}} \ln\left(\frac{\hat{y}_{j\phi}}{\hat{y}_{j\theta}}\right)\hat{y}_{j\phi} + \sum_{x_j \in \mathcal{N}_{x_i}^{training}} \ln\left(\frac{\hat{y}_{j\theta}}{\hat{y}_{j\phi}}\right)\hat{y}_{j\theta}\Big) \tag{4.4}$$

We propose a neighborhood fidelity objective function, $L_{JS_{local}}$, that is calculated using the Jensen-Shannon divergence (JS) as follows:

$$\mathcal{L}_{JS_{local}}\left(\mathcal{N}_{x_{1:N}}^{training}, f_\theta, g_\phi^{global}\right) := D_{\mathrm{JS}_{\mathrm{local}}}(\hat{y}_\phi, \hat{y}_\theta) \tag{4.5}$$

Here, $x_i$ represents an instance and $\mathcal{N}_{x_i}^{training}$ its neighbors, $f()$ denotes the predictor's output, and $g_\phi^{global}()$ denotes the global explainer's prediction. The difference between these two terms measures the variability in the predictions within a local neighborhood.

We propose two Locally Explainable Machine Learning Frameworks, (1) a **Pre-hoc Local Explainability Framework**, see Figure 4.1; and (2) a **Co-hoc Local Explainability Framework**, see Figure 4.2. Both frameworks consist of 2 phases by integrating local neighborhoods to achieve local explainability. Phase 1 (Training for Fidelity) is the training phase that optimizes the agreement between the white-box explainer and black-box models, quantified by fidelity within the neighboring training instances. Phase 2 (Generating Local Explanations) performs fine-tuning of the white-box explainer model within the neighboring training instances closest to a new test instance.

We use the nearest neighborhood algorithm to identify a set of neighboring instances for each local instance in the dataset, such as k-nearest neighbors (k-NN). The intuition behind considering local neighborhoods is that similar inputs are expected to have similar outputs while capturing the model's behavior near each instance by focusing on the local neighborhood.

We compute the predicted probability distributions for each instance and its corresponding neighbors using the predictor model $f_\theta$ and the explainer model $g_\phi^{global}$. This step results in multiple probability distributions for each training instance, representing the predictions of the black-box and white-box explainer models within the local neighborhood. Then, we use the Jensen-Shannon divergence, computed between the predictions of the black-box predictor model and the explainer model for the local in-training neighborhoods

surrounding an instance. This divergence measure quantitatively assesses the consistency between the predictor and explainer models on a local level.

### 4.3.1 PHASE 1: Integrating Local Explainability with Neighbors in Training

In the following, we will modify the loss functions of the Pre-hoc and Co-hoc frameworks proposed in Chapter 3 to incorporate *local* explainability as follows:

1. **Pre-hoc local explainability loss during Training Phase 1:** The loss function $\mathcal{L}_{\text{Local-Pre-hoc-JS}}$ consists of three components: the binary cross-entropy loss $\mathcal{L}_{\text{BCE}}$ for prediction accuracy, the JS-local divergence $D_{\text{JS-local}}$ for local explainability, and an L2 regularization term $\mathcal{L}_2$ for model complexity, as follows:

$$\mathcal{L}_{\text{Local-Pre-hoc}} = \mathcal{L}_{\text{BCE}} + \lambda_1 D_{JS_{local}} + \lambda_2 \mathcal{L}_2 \tag{4.6}$$

   Where $\mathcal{L}_{BCE}$ is the binary cross-entropy loss captures prediction errors, $\lambda_1$ is an explainability regularization coefficient that controls the smoothness of the new representation and the trade-off between explainability and accuracy, while $\lambda_2$ is the coefficient for standard $\mathcal{L}_2$ regularization of model parameters $\theta$ that aims to avoid overfitting and exploding gradients.

2. **Co-hoc local explainability loss during Training Phase 1:** The loss function $\mathcal{L}_{\text{Local-Co-hoc-JS}}$ consists of three components: the binary cross-entropy loss $\mathcal{L}_{\text{f-BCE}}$ for prediction accuracy, $\mathcal{L}_{g^{global}BCE}$ for global explainer accuracy, the JS-local divergence $D_{\text{JS-local}}$ for local explainability, and an L2 regularization term $\mathcal{L}_2$ for model complexity, as follows:

$$\mathcal{L}_{\text{Local-Co-hoc}} = \mathcal{L}_{\text{f-BCE}} + \mathcal{L}_{g^{global}BCE} + \lambda_1 D_{JS_{local}} + \lambda_2 \mathcal{L}_2(f) + \lambda_2 \mathcal{L}_2(g^{global}) \tag{4.7}$$

   Where given input data $\mathcal{X}$ and true outputs $y$, to be explained by an explainer model family $g^{global}$ with parameters $\phi$, $\mathcal{L}_{BCE}$ is the binary cross-entropy loss that ensures accurate predictions, $\lambda_1$ is an explainability regularization coefficient that controls the smoothness of the new representation and the trade-off between explainability and accuracy. At the same time, $\lambda_2$ is the coefficient for standard $\mathcal{L}_2$ regularization of model parameters $\theta$ that aims to avoid overfitting and exploding gradients.

**Algorithm 4.1** Training PHASE 1 for Pre-hoc: Integrating Local Explainability with Neighbors in Training

**Require:** input training instances $X_{train}$, true label $y$, nearest neighborhood function $GetNeighbors()$, number of neighborhood instances $k$, and parameter $\lambda$, the coefficient for the explainability regularization term.

**for** each $x_i$ in $X_{train}$ **do**
  $\mathcal{N}^{training}(x_i) \leftarrow$ GetNeighbors$(x_i, k, X_{train})$ ▷ Get k-NN to training instance $x_i$ from Training set
**end for**
Initialize $\phi$ and $\theta$
**for** each $x_i$ in $X_{train}$ **do**
  Compute $p^{\theta} = f_{\theta}(x_i)$ ▷ Predictions from predictor model
  Compute $p^{\phi} = g_{\phi}^{global}(x_i)$ ▷ Predictions from explainer model
  Compute $\mathcal{L}_{JS_{local}} = ComputeLocalLoss_{JS}(f_{\theta}(x_i), g_{\phi}^{global}(x_i), \mathcal{N}^{training}(x_i))$
  Compute $\mathcal{L}_{g^{global}BCE}$ ▷ White-box model loss
  $\mathcal{L}_{Local-Pre-hoc} = \mathcal{L}_{f-BCE} + \lambda \cdot \mathcal{L}_{JS_{local}}$ ▷ Black-box model loss
  Update $f_{\theta}$ using gradient descent on $\theta$: $\theta \leftarrow \theta - \alpha \nabla_{\theta} \mathcal{L}_{Local-Pre-hoc}$
  Update $g_{\phi}^{global}$ using gradient descent on $\phi$: $\phi \leftarrow \phi - \alpha \nabla_{\phi} \mathcal{L}_{g^{global}BCE}$
**end for**
Return $f_{\theta}$ and $g_{\phi}^{global}$
**procedure** $ComputeLocalLoss_{JS}(f_{\theta}, g_{\phi}^{global}, \mathcal{D})$ ▷ Compute average JS divergence for input subset $\mathcal{D}$
  $\mathcal{L}_{\text{JS}} \leftarrow 0$
  **for** All $x_i \in \mathcal{D}$ **do**
    $\hat{y}_{\phi}^{(i)} \leftarrow g_{\phi}^{global}(x_i)$
    $\hat{y}_{\theta}^{(i)} \leftarrow f_{\theta}(x_i)$
    $\mathcal{L}_{JS_{local}}^{(i)} \leftarrow D_{JS_{local}}(\hat{y}_{\theta}^{(i)}, \hat{y}_{\phi}^{(i)})$
    $\mathcal{L}_{JS_{local}} \leftarrow \mathcal{L}_{JS_{local}} + \mathcal{L}_{JS_{local}}^{(i)}$
  **end for**
  $\mathcal{L}_{JS_{local}} \leftarrow \frac{1}{|\mathcal{D}|} \mathcal{L}_{JS_{local}}$
  **return** $\mathcal{L}_{JS_{local}}$
**end procedure**

Algorithm 4.2 presents the method for integrating the Jensen-Shannon divergence for local explainability within neighborhoods during the training of the black-box model. The algorithm takes as input the black-box model $f_{\theta}$, the white-box model $g_{\phi}^{global}$, the training dataset $X_{train}$ with their true labels $y$, and the hyperparameter $\lambda_1$. It also requires the nearest neighborhood function $GetNeighbors$ and the number of neighborhood instances $k$.

The $ComputeLocal_{JS}$ procedure calculates the local JS divergence between the predictions of the black-box model $f_{\theta}$ and the white-box model $g_{\phi}^{global}$ for each training instance in the input data $X$ and its corresponding neighbors. It retrieves the neighbors $\mathcal{N}^{training}(x_i)$

**Algorithm 4.2** Training PHASE 1 for Co-hoc: Integrating Local Explainability with Neighbors in Training

**Require:** input training instances $X_{train}$ with their true labels $y$, nearest neighborhood function $GetNeighbors()$, number of neighborhood instances $k$, and explainability regularization coefficient $\lambda$.

**for** each $x_i$ in $X_{train}$ **do**
$\mathcal{N}^{training}(x_i) \leftarrow \text{GetNeighbors}(x_i, k, X_{train})$ ▷ Get k-NN of training instance $x_i$ from Training set
**end for**
Initialize $\phi$ and $\theta$
**for** each $x_i$ in $X_{train}$ **do**
Compute $p_\theta = f_\theta(x_i)$ ▷ Predictions from predictor model
Compute $p_\phi = g_\phi^{global}(x_i)$ ▷ Predictions from explainer model
Compute $\mathcal{L}_{JS_{local}} = ComputeLocalLoss_{JS}(f_\theta(x_i), g_\phi^{global}(x_i), \mathcal{N}^{training}(x_i))$
$\mathcal{L}_{Local-Co-hoc} = \mathcal{L}_{f-BCE} + \mathcal{L}_{g^{global}BCE} + \lambda \cdot \mathcal{L}_{JS_{local}}$
Update $f_\theta$ and $g_\phi^{global}$ using gradient descent on $\theta$: $\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}_{Local-Co-hoc}$
**end for**
Return $f_\theta$ and $g_\phi^{global}$
**procedure** $ComputeLocalLoss_{JS}(f_\theta, g_\phi^{global}, \mathcal{D})$ ▷ Compute average JS divergence for input subset $\mathcal{D}$
$\mathcal{L}_{\text{JS}} \leftarrow 0$
**for** All $x_i \in \mathcal{D}$ **do**
$\hat{y}_\phi^{(i)} \leftarrow g_\phi^{global}(x_i)$
$\hat{y}_\theta^{(i)} \leftarrow f_\theta(x_i)$
$\mathcal{L}_{JS_{local}}^{(i)} \leftarrow D_{JS_{local}}(\hat{y}_\theta^{(i)}, \hat{y}_\phi^{(i)})$
$\mathcal{L}_{JS_{local}} \leftarrow \mathcal{L}_{JS_{local}} + \mathcal{L}_{JS_{local}}^{(i)}$
**end for**
$\mathcal{L}_{JS_{local}} \leftarrow \frac{1}{|\mathcal{D}|}\mathcal{L}_{JS_{local}}$
**return** $\mathcal{L}_{JS_{local}}$
**end procedure**

of each instance $x_i$ using the $GetNeighbors$ function, computes the predictions of both predictor and explainer models for the neighbors, and calculates the average JS divergence, $\mathcal{L}_{JS}$, between these predictions.

The training phase iterates over each training instance $x_i$ from dataset $X_{train}$. For each instance, it computes the black-box model's prediction $p_\theta$ and the local JS divergence $L_{JS}$ using the $ComputeLocal_{JS}$ procedure. The binary cross-entropy loss $L_{BCE}$ is also calculated between the black-box model's prediction and the true label $y$. The total loss $L_{total}$ is then computed as a weighted sum of $L_{BCE}$ and $L_{JS}$, where the weight $\lambda_1$ controls the importance of local explainability. Finally, the black-box model's parameters $\theta$ are

updated using gradient descent based on the total loss.

By incorporating explainability within neighborhoods during the training process, the proposed framework aims to generate an explainer model that is faithful to the model's behavior and consistent with the explanations provided by the interpretable white-box model in the local neighborhood of each instance. Up to this point, although the training strives to learn a predictor model close to the explainer model within local neighborhoods, the explainer model is still considered global. In the next phase, a distinct local explainer model will be computed after the prediction has been computed for a new (test) instance. However, unlike traditional post-hoc surrogate explainer models, which are trained from scratch on a neighborhood, the proposed explainer model will be obtained by *fine-tuning* the global model that had already been pre-trained in Phase 1 during training,to cut on the explainer training cost after prediction.

### 4.3.2 PHASE 2: Generating Local Explanations

**Algorithm 4.3** Testing PHASE 2: Computing Local Explanations

**Require:** White-box model $g_{\phi}^{global}$, input training instances $X_{train}$ with their true labels $y$, nearest neighborhood function $GetNeighbors()$, number of neighborhood instances $k$, testing instance $x_i$

$\mathcal{N}^{testing}(x_i) \leftarrow$ GetNeighbors$(x_i, k, X_{train})$ ▷ Get k-NN to training instance $x_i$ from Training set
Compute $p_{\phi} = g_{\phi}^{global}(\mathcal{N}^{testing}(x_i))$ ▷ Predictions from explainer model
**for** All $x_j \in \mathcal{N}^{testing}(x_i)$ **do** ▷ Get predictor outputs for the local training neighbors
  $\hat{y}_{\phi_i,j}^{local} \leftarrow g_{\phi}^{global}(x_j)$
**end for**
$\phi_i \leftarrow \phi$ ▷ initialize local model to the global model
**for** t=1 to $T_{finetune}$ **do**
  **for** All $x_j \in \mathcal{N}^{testing}(x_i)$ **do**
    $\hat{y}_{\phi_i,j}^{local} \leftarrow g_{\phi_i}^{local}(x_j)$ ▷ Get local explainer outputs for the local training neighbors
  **end for**
  Update Local Explainer Loss $\mathcal{L}_{\text{Local-Explainer}}$ using Eq. 4.8
  $\phi_i \leftarrow \phi_i - \alpha \nabla_{\phi_i} \mathcal{L}_{\text{Local-Explainer}}$ ▷ Update $g_{\phi_i}^{local}$ using gradient descent
**end for**
Extract feature importances feature_importances from $g_{\phi_i}^{local}$ using Algorithm 3.3 and the set of features in the data
**return** feature_importances

In Phase 2 of our locally explainable machine learning framework, we focus on generating local explanations by fine-tuning the global white-box explainer model $g_{\phi}^{global}$, pre-

viously trained (pre-trained) in Phase 1, for each new individual testing instance $x_i$ to be predicted and explained. The fine-tuned white-box model $g_{\phi_i}^{local}$ is a sparse linear regression model.

To fine-tune the global white-box explainer model previously learned in Phase 1, we first identify the nearest training set neighbors relative to the test instance we want to explain using the $GetNeighbors$ function. These neighboring instances are called in-training neighborhood instances.

$$\mathcal{N}^{testing}(x_i) = \text{GetNeighbors}(x_i, k, X_{train})$$

These neighboring testing instances can provide a local context to transfer our previously constructed global explanations of the black box model's behavior to be focused on the specific test instance. Hence, we use the in-testing neighborhood instances to fine-tune the pre-trained global white-box explainer model $g_{\phi}^{global}$. The fine-tuning process involves performing a few iterations of optimizing the global explainer model's $g_{\phi}^{global}$ parameters to minimize the white-box model loss function, regularized using the Jensen-Shannon divergence loss between the fine-tuned explainer's prediction $\hat{y}_{\phi_i}^{local} = g_{\phi_i}^{local}(x_i)$ and the black box predictions $\hat{y}_{\theta} = f_{\theta}(x_i)$ over the in-testing neighborhood. Specifically, the loss function is given by

$$\mathcal{L}_{\text{Local-Explainer}} = D_{\text{JS}_{\text{local}}}^{expl}(\hat{y}_{\phi_i}^{local}, \hat{y}_{\theta}) \tag{4.8}$$

where

$$\begin{aligned} D_{\text{JS}_{\text{local}}}^{expl}(\hat{y}_{\phi_i}^{local}, \hat{y}_{\theta}) = \frac{1}{2}\Bigg( & \sum_{x_j \in \mathcal{N}^{training}(x_i)} \ln\left(\frac{\hat{y}_{\phi_i}^{local}}{\hat{y}_{j\theta}}\right) \hat{y}_{\phi_i}^{local} \\ & + \sum_{x_j \in \mathcal{N}^{testing}(x_i)} \ln\left(\frac{\hat{y}_{j\theta}}{\hat{y}_{\phi_i}^{local}}\right) \hat{y}_{j\theta}\Bigg) \end{aligned} \tag{4.9}$$

Once fine-tuning is finished, the sparse linear model $g_{\phi_i}^{local}$ will finally be used to calculate feature importance scores using Algorithm 3.3 in Chapter 3, which will quantify the contribution of each feature to the model's predictions. The feature importance scores quantify the relative importance of each feature to the model decision-making process for a given test instance, hence serving as a local explanation to help users understand the factors influencing the model's predictions for a specific test instance.

## 4.4 Comparison to Classical Post-hoc Explainability Methods such as LIME

Our proposed in-training explainability framework ingrains the explainability within the model training phase by using the Jensen-Shannon (JS) divergence for local fidelity. In contrast, traditional post-hoc methods are generally model-agnostic techniques that seek to interpret **already trained** black box models. These methods usually generate explanations by either approximating the behavior of complex black box models using interpretable and simple surrogate models as in the case of LIME [17] or by computing the contribution of the data features to the predictions without learning a surrogate model, as in the case of SHAP [18]. Although these methods can be applied to any trained model, they often involve additional computational overhead at explanation time after prediction. They may also generate explanations inconsistent with the model because, by their definition and nature (being model-agnostic), they function entirely outside the scope of the model and its training. Our in-training framework addresses this limitation by learning explainer models tightly coupled with the black model model during training. The black model is influenced by the explainer, and in addition, in the case of the Co-hoc method, the explainer model is directly influenced by the black box model.

Although both post hoc methods and our proposed in-training explainability framework target local explainability, they adopt different principles. For example, LIME tries to *approximate* the black-box model using an interpretable surrogate in a post-hoc manner. In contrast, our proposed in-training explainability framework aims to *align* the black-box model's decisions with an interpretable explainer model during training. TTherefore, theintraining explainability framework can,be seen as a proactive machine learning method to ensure local explainability. At the same time, LIME is considered instead as a reactive method that attempts to interpret decisions after the fact.

Furthermore, our local in-training explainability framework works by learning a predictive model that is tightly coupled with the explainer during training, followed by *fine-tuning* the explainers to the local neighborhood of a new instance after prediction in a post-hoc manner however this is designed using a pre-training of the explainers alongside the predictor model during the training phase, followed by a lighter fine-tuning of the explainer in the post-hoc phase, hence alleviating the costly task of learning explainers from scratch for each new instance. Depending on the given real-time demands and budgets re-

garding computational resources, desired robustness of explanations, or flexibility in model training, one approach might be favored.

## 4.5 Experimental Results

### 4.5.1 Research Questions

We compare the proposed local explainability methods with a post-hoc method, LIME [17], by answering the following research questions.

- **RQ 4.1 - Explanatory Power**: How well does the explainer model mimic the predictor model in Phase 1?
- **RQ 4.2 - Trade-off**: How does the $\lambda$ value affect the accuracy and fidelity score in Phase 1? **RQ 4.3 - Framework Comparison:** What are the differences between the Pre-hoc and Co-hoc frameworks in Phase 1?
- **RQ 4.4 - Locality and Stability:** How well do the explanations capture the local behavior of the model compared to LIME in Phase 2?
- **RQ 4.5 - Neighborhood Size:** How does neighborhood size affect neighborhood fidelity, stability, and computational cost in Phase 2?
- **RQ 4.6 - Computational Efficiency:** What is the computational cost of generating explanations for our proposed frameworks and the LIME post hoc method in Phase 2?

### 4.5.2 Datasets

We experimented with three publicly accessible real-world datasets. All three datasets were used in a binary classification setting.

- HELLOC

  FICO HELOC data set [115] contains 10,459 anonymized information on the home equity line of credit (HELOC) applications made by real homeowners. The target variable is the risk performance, which predicts whether the homeowner qualifies for a line of credit.

- Movielens 100k

  Movielens has 100k movie ratings and 100,000 ratings based on 1000 users on 1700 movies. MovieLens has 1M movie ratings and 1 million ratings based on 6000 users of 4000 movies. For Movielens datasets [114], the classification target is the movie rating. We aim to learn whether a user will like or dislike a movie. Hence, the target is discretized into liked and disliked, with 1 being the class label for a rating of 3 and above and 0 being the class label for a rating of less than 3.

- Grasping Dataset

  We evaluate the performance of our approach on the grasping dataset obtained by simulation using the Smart Grasping Sandbox [119]. The public dataset was created to investigate the effectiveness of using machine learning to predict whether a robot's grasp is stable while grasping a ball. The data set has been annotated with an objective grasp of consistency. It contains data obtained from the three joints on each of the grasper's three fingers' (position, velocity, and effort), amounting to about 54,000 unique data points and 29 measurements for each experiment.

  The classification target is the predicted grasp robustness. Moreover, the output is discretized as 1 for a stable grasp and 0 for an unstable grasp. A grasp is considered stable if the robustness value is more than 100.

### 4.5.3 Evaluation Protocols

To assess the classification accuracy, we use the Area under the ROC Curve, AUC($f_\theta$, $\hat{y}$). Each dataset is split randomly into training, validation, and test sets in the ratio 80:10:10. After training on every batch with a learning rate of 0.001, AUC is calculated on the validation and test datasets. We measure all the metrics on a held-out test set. All measurements are repeated five times, and the results are averaged across runs and tabulated along with the standard deviation. All models are trained with $L_2$ regularization until validation accuracy is stabilized for at least ten epochs. For local explainabilty neighborhood generation, we set the number of neighbors $k = 10$ during Training Phase 1 and $k = 100$ during Testing Phase 2 (after trial and error). All the models were implemented using the PyTorch framework [120] and executed on an NVIDIA Tesla V100 GPU, 16 GB RAM and CPU Intel(R) Xeon(R) CPU, 2.20GHz, 13 GB RAM.

We use **fidelity**, also known as descriptive accuracy [116], to measure how accurately an explanation method can mimic the behavior of a black-box classifier in terms of assigning class labels to data records. Specifically, we use AUC($f_\theta$, $g_\phi$) to estimate the fidelity. Our baseline for fidelity is the AUC of the original black-box predictor model and the explainer model, which can also be considered a post-hoc explainability score without any optimization. To evaluate our suggested models against the LIME post-hoc method [17], we employ the metrics comprehensively described in Chapter 2 in Sections 2.6, namely **point fidelity** [17], **neighborhood fidelity** [68], and **stability** [27].

Another crucial factor to consider when evaluating the practicality and scalability of explanation methods is the **computational cost** of generating explanations using our proposed explainable frameworks (EF) compared to the LIME state-of-the-art post-hoc method. More specifically, we evaluated the efficiency of both methods in terms of *both* **(1)** the *training* time and **(2)** the time required to generate local *explanations* for individual test instances.

The difference between the total *training* time of the regularized explainable model in the Explainable Framework (EF) and the original black-box model (BB) gives us the additional computation time introduced by the explainability regularization. This additional computation time quantifies the computational overhead of incorporating explainability regularization into the training process of the EF compared to the unregularized black-box model (BB). This additional computation time helps us assess the trade-off between the improved explainability provided by our proposed framework and the increased computational cost associated with the explainability regularization. This information can be valuable for assessing the practical implications of using EF in real-world scenarios, where computational resources may be limited, and the balance between explainability and efficiency is crucial.

We also measure the time required to *explain* individual instances using EF and LIME to assess the added cost of *generating explanations*. The computational cost is calculated by measuring the average time to generate explanations for individual instances using EF or LIME. By comparing the average explanation times and standard deviations, we can assess the computational efficiency of our explainable framework in generating explanations for individual instances. Finally, the *total* computational cost, which includes training *and* explanation times, comprehensively compares the overall efficiency of all methods.

### 4.5.4 Baselines

We compared our Pre-hoc and Co-hoc locally explainable predictor models with their original non-regularized black-box (BB) version, Factorization Machines [117], sstate-of-the-artmodel for classification, regression, and recommendation tasks. The explainer white-box model (WB) is a sparse logistic regression model, which is inherently explainable,and thus provides the explanation.

### 4.5.5 Parameter Settings

We implemented our proposed methods based on PyTorch. All models are learned by optimizing binary cross-entropy with Adam [118], an extension of stochastic gradient descent. The batch size is selected as 64, 2056, and 64, respectively, for the ML-100K, Grasping, and HELOC datasets, all obtained by validation as the optimal batch size for accuracy for their respective datasets. We tested for $\lambda_1$ in $\{0.01, 0.05, 0.25, 0.5, 1\}$. The regularization weight of the loss function is estimated using a mini-batch. We pick the best regularization weight for each dataset using the validation set and use that for the final evaluation. The final evaluation is done by retraining the models using their chosen configurations and evaluating them on the test set.

### 4.5.6 Results and Discussion

**RQ 4.1 - Explanatory Power: How well does the explainer model mimic the predictor model in Phase 1?**

To assess the global explanatory power of the Pre-hoc and Co-hoc frameworks, we examine how well the explanations generated by these frameworks capture the nuances of the black-box model's decision-making process by analyzing the fidelity AUC scores,since they measure the agreement between the predictions of the predictor model and the global explainer model, for different values of the explainability regularization parameter $\lambda$.

Figure 4.6b (b) presents the fidelity AUC scores for the Pre-hoc framework on the Grasping dataset for different values of $\lambda$. As $\lambda$ increases, the fidelity AUC shows a notable improvement, increasing from around 0.80 at $\lambda = 0.01$ to approximately 0.95 at $\lambda = 1$. This indicates that higher values of $\lambda$ strengthen the regularization effect, encouraging the Pre-hoc Predictor to align more closely with the WB Explainer. Since the WB Explainer

is used to guide the learning of the Pre-hoc Predictor, the high fidelity scores suggest that the explanations generated by the Pre-hoc framework effectively capture the nuances of the black-box model's decision-making process.

Similarly, Figure 4.4b (b) presents the fidelity AUC scores for the Co-hoc framework in the HELOC data set for different values of $\lambda$. As $\lambda$ increases, the fidelity AUC ssignificantly improves increasing from around 0.78 at $\lambda = 0.01$ to approximately 0.96 at $\lambda = 1$. This suggests that the Co-hoc framework also effectively captures the nuances of the black-box model decision-making process, with higher values of $\lambda$ leading to better alignment between the Co-hoc Predictor and the WB Explainer.

Comparing the fidelity AUC scores of the Pre-hoc and Co-hoc frameworks, we observe that both frameworks achieve high fidelity scores, indicating their ability to generate explanations that closely capture the decision-making process of the black-box model. The high fidelity scores result from the regularization term in the objective function, which encourages the black-box predictor to align with the WB Explainer. By minimizing the divergence between the predictor's and explainer's outputs, the generated explanations effectively capture the nuances of the black-box model's decisions. Furthermore, the Co-hoc framework exhibits slightly higher fidelity scores than the Pre-hoc framework for the same values of $\lambda$. This can be attributed to the joint optimization process in the Co-hoc framework, where the black-box predictor and the WB Explainer are trained simultaneously. Joint optimization allows for a stronger alignment between the predictor and the explainer, resulting in explanations that better capture the nuances of the black-box model decision-making process.

To summarize, tanalyzingthe fidelity AUC scores for the Pre-hoc and Co-hoc frameworks on the Grasping, HELOC, and ML-100k datasets highlights their ability to generate explanations that effectively capture the nuances of the black-box model's decision-making process. The high fidelity scores achieved by both frameworks demonstrate the effectiveness of using the WB Explainer to guide the learning of the black-box predictor and generate explanations that align closely with the model's decisions. The Co-hoc framework's joint optimization approach leads to slightly higher fidelity scores than the Pre-hoc framework, indicating a stronger alignment between the predictor and the explainer in capturing the model's decision-making process.

**RQ 4.2 - Trade-off between Accuracy and Explanation Fidelity**: How does explainability regularization $\lambda$ affect the accuracy and fidelity score in Phase 1?

To analyze the effect of the explainability regularization parameter $\lambda$ on the accuracy and fidelity scores, we examine Figures 4.3, 4.4, 4.5, 4.6, 4.8, 4.7.

Figure 4.3a shows the accuracy AUC scores for the Pre-hoc Predictor, BB Predictor, and WB Explainer models in the HELOC dataset for different values of $\lambda$. As $\lambda$ increases from 0.01 to 1, the accuracy AUC of the Pre-hoc Predictor remains relatively stable, with a slight increase from around 0.744 to 0.746. The accuracy of the Pre-hoc Predictor is consistently higher than that of the WB Explainer, indicating that the Pre-hoc framework maintains a good balance between accuracy and explainability. The BB Predictor, which is the unregularized black-box model, has a slightly higher accuracy than the Pre-hoc Predictor, but the difference is marginal.

Figure 4.3b (b) presents the fidelity AUC scores for the Pre-hoc framework on the HELOC dataset for different values of $\lambda$. The fidelity AUC measures the agreement between the predictions of the Pre-hoc Predictor and the WB Explainer. As $\lambda$ increases, the fidelity AUC ssignificantly improves rising from around 0.78 at $\lambda = 0.01$ to approximately 0.95 at $\lambda = 1$. This confirms that higher values of $\lambda$ strengthen the regularization effect, encouraging the Pre-hoc Predictor to align more closely with the WB Explainer, thereby enhancing the explainability of the model.

The results demonstrate that the explainability regularization parameter $\lambda$ plays a crucial role in controlling the trade-off between accuracy and explainability in the Pre-hoc framework. Lower values of $\lambda$ prioritize precision, whereas higher values emphasize explainability. The choice of $\lambda$ depends on the specific requirements of the application and the desired balance between accuracy and interpretability. Even with high values of $\lambda$, the accuracy of the Pre-hoc Predictor remains competitive with the BB Predictor, indicating that the Pre-hoc framework effectively incorporates explainability without significantly sacrificing predictive performance.

Analyzing the accuracy AUC scores for the Co-hoc framework on the HELOC dataset, in Figure 4.4a, we observe an interesting behavior when the explainability regularization parameter $\lambda$ is set to 1. At this high regularization level the accuracy of both the Co-hoc Predictor and the WB Explainer decreases and falls below their previous results. This phenomenon is not observed in the Pre-hoc framework, where accuracy remains relatively stable for the WB explainer even at high values of $\lambda$. Furthermore, the regularized Co-hoc BB accuracy exceeds that of the BB baseline only within a narrower range of $\lambda$ compared to

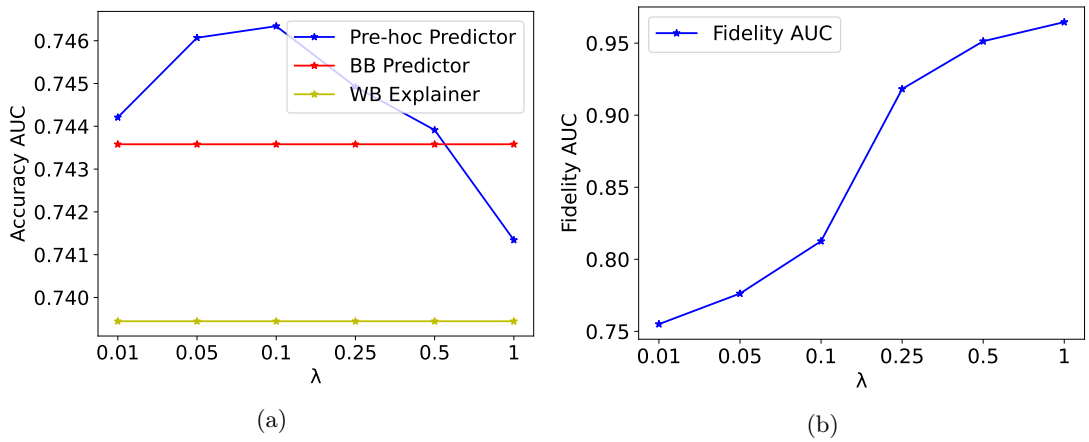



Figure 4.3: HELOC Dataset. Pre-hoc Explainability Framework Comparison in Accuracy AUC and Fidelity AUC for different explainability regularization $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Pre-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model.

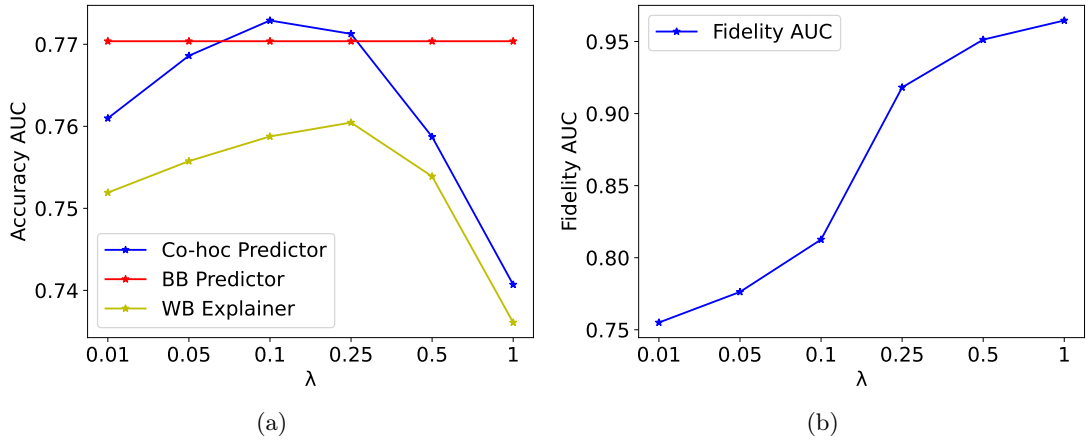



Figure 4.4: HELOC Dataset. Co-hoc Explainability Framework (a) Accuracy AUC (b) Fidelity AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Co-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model.

the Pre-Hoc case. This behavior can be attributed to the joint optimization process in the Co-hoc framework. In the Co-hoc setting, the black-box model and the white-box explainer are trained simultaneously, with the objective of minimizing the combined loss function that includes the explainability regularization term. When $\lambda = 1$, the regularization term dominates the loss function, leading to a strong emphasis on aligning the predictions of the Co-hoc Predictor with those of the WB Explainer. As a result, the Co-hoc Predictor may overfit the WB Explainer, sacrificing its predictive performance.

Moreover, due to the joint optimization in the Co-hoc framework, the accuracy of the

WB Explainer and the Co-hoc Predictor also decreases. This is because the WB Explainer is actively involved in the training process and is influenced by the regularization term. The strong regularization at $\lambda = 1$ forces the WB Explainer to adapt its predictions to align with the Co-hoc Predictor, potentially decreasing its accuracy.

In contrast, in the Pre-hoc framework, the WB Explainer is not regularized or modified during the training process. Instead, it is used as a fixed reference model to regularize the black-box model. As a result, the accuracy of the WB Explainer remains constant across different values of $\lambda$ in the Pre-hoc setting. The Pre-hoc framework focuses on aligning the predictions of the black-box model with those of the fixed WB Explainer without affecting the explainer's performance.

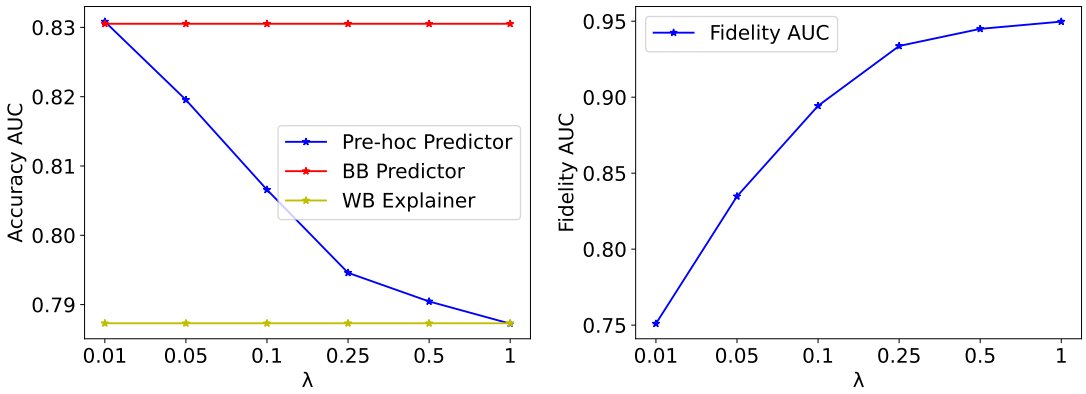



Figure 4.5: Grasping Dataset. Pre-hoc Local Explainability Framework (a) Accuracy Fidelity (b) Fidelity AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Pre-hoc Local Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model.

Figure 4.6a displays the accuracy AUC scores for the Pre-hoc Local Predictor, BB Predictor, and WB Explainer models on the Grasping dataset for different values of $\lambda$. As $\lambda$ increases from 0.01 to 1, the accuracy AUC of the Pre-hoc Predictor remains relatively stable, with a slight decrease from around 0.83 to 0.81. The accuracy of the Pre-hoc Predictor is consistently higher than that of the WB Explainer, demonstrating that the Pre-hoc framework maintains a good balance between accuracy and explainability. The BB Predictor, which is the unregularized black-box model, has a slightly higher accuracy than the Pre-hoc Predictor, but the difference is minimal.

Figure 4.6b (b) presents the fidelity AUC scores for the Pre-hoc framework in the

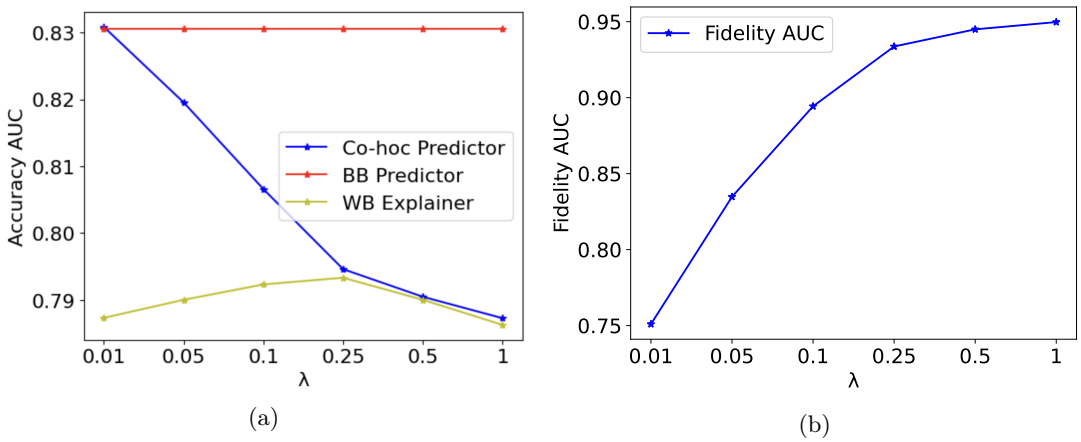



Figure 4.6: Grasping Dataset. Co-hoc Local Explainability Framework (a) Accuracy Fidelity (b) Fidelity AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Co-hoc Local Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model.

Grasping dataset for different values of $\lambda$. The fidelity AUC measures the agreement between the predictions of the Pre-hoc Predictor and the WB Explainer. As $\lambda$ increases, the fidelity AUC shows a notable improvement, increasing from around 0.80 at $\lambda = 0.01$ to approximately 0.95 at $\lambda = 1$. This indicates that higher values of $\lambda$ strengthen the regularization effect, encouraging the Pre-hoc Predictor to align more closely with the WB Explainer, thereby enhancing the explainability of the model.

The results for the Grasping dataset are consistent with the findings of the HELOC dataset, demonstrating the effectiveness of the explainability regularization parameter $\lambda$ in controlling the trade-off between accuracy and explainability in the Pre-hoc framework. Lower values of $\lambda$ prioritize precision, whereas higher values emphasize explainability. The choice of $\lambda$ depends on the specific requirements of the application and the desired balance between accuracy and interpretability. With high values of $\lambda$, the accuracy of the Pre-hoc Predictor remains competitive with that of the BB Predictor on the Grasping dataset, indicating that the Pre-hoc framework effectively incorporates explainability without significantly compromising predictive performance.

The analysis of the Grasping and ML-100k datasets reinforces the findings from the HELOC dataset, highlighting the effectiveness of the explainability regularization parameter $\lambda$ in controlling the trade-off between accuracy and explainability in the proposed frameworks. Higher values of $\lambda$ lead to improved fidelity between the BB predictor and the WB

Explainer, enhancing the interpretability of the model while maintaining a good level of accuracy. The results demonstrate the versatility of the proposed frameworks in achieving a balance between accuracy and explainability across different datasets, providing users with the flexibility to adjust the model based on their specific needs and requirements.

In summary, the explainability regularization parameter $\lambda$ allows users to control the trade-off between accuracy and explainability in the Pre-hoc framework. Higher values of $\lambda$ lead to improved fidelity between the Pre-hoc Predictor and the WB Explainer, enhancing the interpretability of the model while maintaining good accuracy. The results highlight the effectiveness of the proposed frameworks in achieving a balance between accuracy and explainability, providing users with the flexibility to adjust the model based on their specific needs and requirements.

**RQ 4.3 - Framework Comparison: What are the differences between the Pre-hoc and Co-hoc frameworks in Phase 1?**

The observations in RQ 4.1 and RQ 4.2 highlight the differences between the Co-hoc and Pre-hoc frameworks in terms of their optimization processes and the impact of regularization on the models involved. Joint optimization of the Co-hoc framework allows for a stronger alignment between the Co-hoc Predictor and the WB Explainer. Still, it may lead to over-regularization and decreased accuracy when $\lambda$ is set too high. On the other hand, the Pre-hoc framework provides a more stable performance across different regularization strengths, as it only regularizes the black-box model while keeping the WB Explainer fixed.

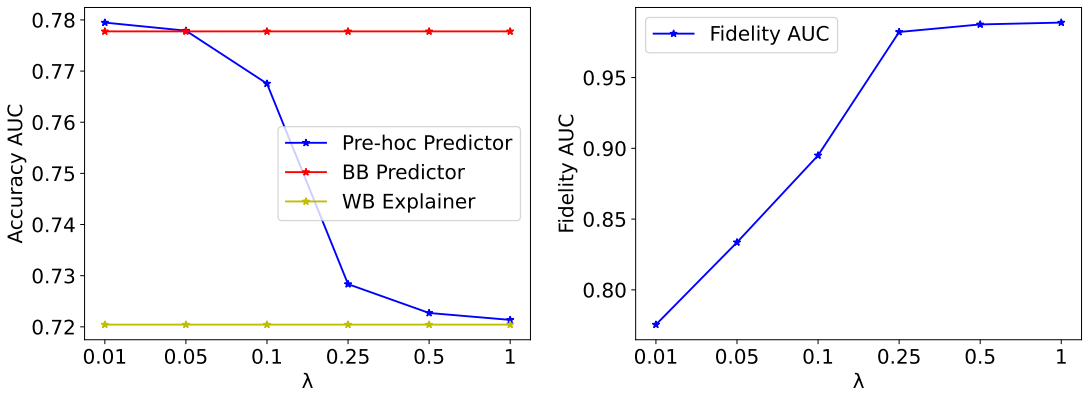



Figure 4.7: ML-100k Dataset. Pre-hoc Explainability Framework Comparison in Accuracy AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Pre-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model.

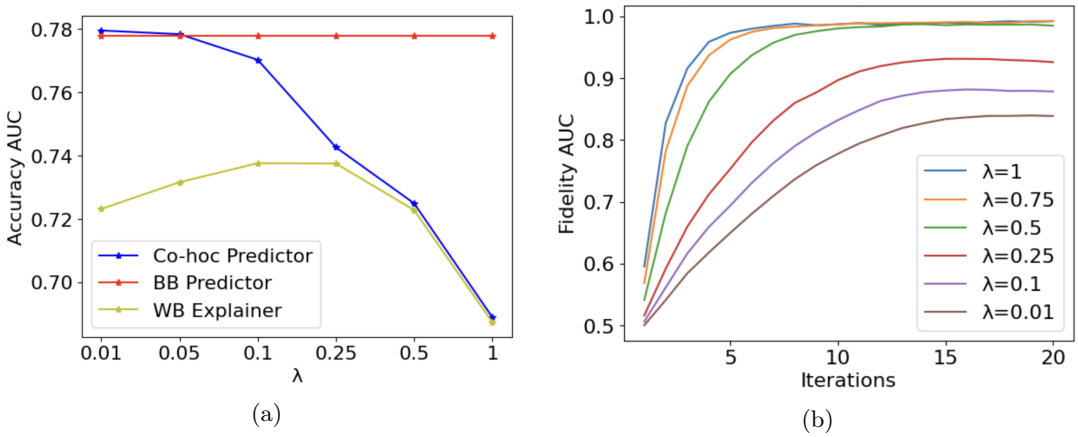



Figure 4.8: ML-100k Dataset. Co-hoc Explainability Framework Comparison in Accuracy AUC for different $\lambda_1$=0.01, 0.05, 0.25, 0.5, 1. Co-hoc Predictor is our proposed model, BB is the original black-box predictor model, and WB is the explainer model.

Another advantage of our Co-hoc framework is that even the white-box model that is learned to explain the black-box predictor achieves a significantly higher prediction accuracy after the Co-hoc learning. This means that the proposed Co-hoc in-training approach can also improve white-box models, which are essential and required in certain high-risk and regulated application tasks in health care and legal decision-making.

When choosing between the Co-hoc and Pre-hoc frameworks, it is essential to consider the specific requirements and priorities of the application. If a very high level of alignment between the predictor and the explainer is desired, and a slight decrease in accuracy is acceptable, the Co-hoc framework may be preferred. However, if maintaining the predictor's accuracy is critical and the explainer's performance is less important, the Pre-hoc framework may be a more suitable choice.

The analysis of the over-regularization behavior in the Co-hoc framework and the comparison with the Pre-hoc framework highlight the trade-offs and considerations involved in selecting the appropriate framework for a given application. The Co-hoc framework's joint optimization allows for a stronger alignment between the models. Still, it may lead to over-regularization, while the Pre-hoc framework provides a more stable performance by keeping the explainer fixed during the regularization process.

**RQ 4.4 - Locality: How well do the explanations capture the local behavior of the model compared to LIME in Phase 2?**

To assess the local explainability of our proposed frameworks, we compare their

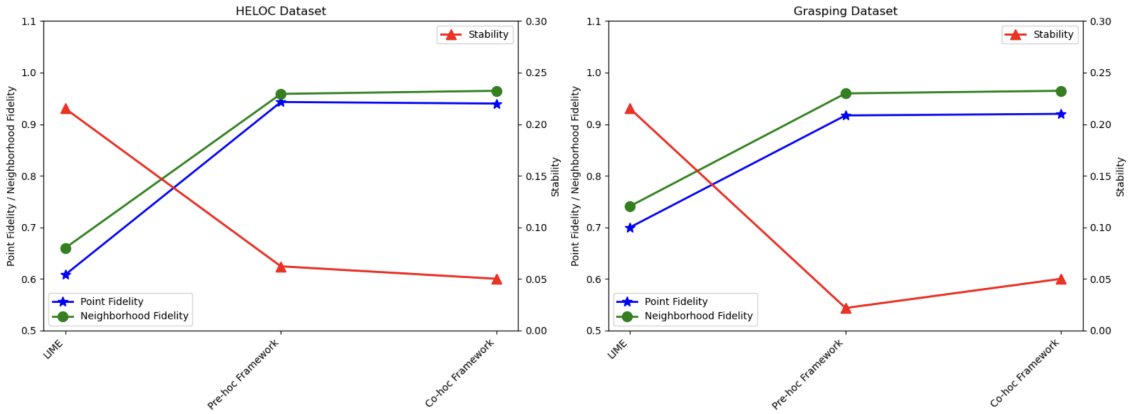



Figure 4.9: Grasping and HELOC Datasets results. Comparison with LIME based on neighborhood fidelity and stability results. $\lambda = 0.25$, $k = 10$. One hundred test instances in the neighborhood, an average of 5 runs.

performance with the LIME post-hoc explainability method [17] in terms of point fidelity, neighborhood fidelity, and stability. Point fidelity measures the agreement between the explanations and predictions for individual instances, while neighborhood fidelity extends this concept to consider the agreement within local neighborhoods around each instance. Stability measures the consistency of explanations across different runs.

Table 4.2 and Figure 4.9 present the HELOC dataset's explanation metrics of our framework and LIME on the held-out test. Regarding Average Neighborhood Fidelity, LIME explanations to the black-box model, the average neighborhood fidelity is 0.6600 with a standard deviation of 0.1939. This result suggests that LIME explanations moderately agree with the black-box model's predictions in the local neighborhoods. White-box (WB) explanations to Pre-hoc explainability framework, the average neighborhood fidelity is 0.9587 with a standard deviation of 0.0766. This indicates a high level of agreement between the white-box explanations and our framework's predictions in the local neighborhoods.

TABLE 4.2

HELOC Dataset results. Comparison with LIME based on neighborhood fidelity and stability results. $\lambda = 0.25$, $k = 10$.

| Explanation Method | Point Fidelity ↑ | Neighborhood Fidelity ↑ | Stability ↓ |
|---|---|---|---|
| LIME | 0.6083 ± 0.0050 | 0.6600 ± 0.1939 | 0.2152 ± 0.0175 |
| Pre-hoc Framework | 0.8270 ± 0.0260 | 0.9587 ± 0.0766 | 0.0623 ± 0.0110 |
| Co-hoc Framework | 0.8300 ± 0.0240 | 0.9647 ± 0.0575 | 0.0502 ± 0.0087 |

Table 4.3 and Figure 4.9 show the Grasping dataset's comparison results. Similar to the HELOC dataset, our Pre-hoc and Co-hoc frameworks outperform LIME in terms of point fidelity, neighborhood fidelity, and stability. The Pre-hoc framework achieves a point fidelity of 0.8270 and a neighborhood fidelity of 0.9597, while the Co-hoc framework achieves scores of 0.8300 and 0.9647, respectively.

TABLE 4.3

Grasping Dataset results. Comparison with LIME based on neighborhood fidelity and stability results. $\lambda = 0.25$, $k = 10$. One hundred test instances in the neighborhood, an average of 5 runs.

| Explanation Method | Point Fidelity ↑ | Neighborhood Fidelity ↑ | Stability ↓ |
|---|---|---|---|
| LIME | 0.7000 ± 0.0150 | 0.7410 ± 0.1345 | 0.2152 ± 0.1667 |
| Pre-hoc Framework | 0.9170 ± 0.0454 | 0.9597 ± 0.0493 | 0.0219 ± 0.0110 |
| Co-hoc Framework | 0.9200 ± 0.0240 | 0.9647 ± 0.0575 | 0.0502 ± 0.0101 |

Table 4.4 and Figure 4.10 presents the results of our Pre-hoc framework with different values of the explainability regularization parameter $\lambda$ on the ML-100k dataset. As $\lambda$ increases, we observe a consistent improvement in point fidelity, neighborhood fidelity, and stability. Without regularization ($\lambda = 0$), the point fidelity is 0.8183, and the neighborhood fidelity is 0.8050. As $\lambda$ increases to 1, the point fidelity reaches 0.9951, and the neighborhood fidelity improves to 0.9953. This indicates that stronger regularization enhances the local explainability of the Pre-hoc framework. Moreover, the stability of the explanations improves with increasing $\lambda$, as evidenced by the decreasing values in the stability column.

The results in all three data sets consistently demonstrate significantly higher scores in all three metrics than LIME (p-value $< .05$), thus showing our proposed frameworks' effectiveness in capturing the model's local behavior. The high point fidelity and neighborhood fidelity scores indicate that our frameworks generate explanations consistent with the model's predictions for individual instances and within local neighborhoods. Improved stability scores suggest that our frameworks produce more consistent explanations across different runs than LIME.

It is worth noting that the Co-hoc framework consistently achieves slightly higher neighborhood fidelity scores than the Pre-hoc framework. This can be attributed to the joint optimization process in the Co-hoc framework, where the black-box model and the

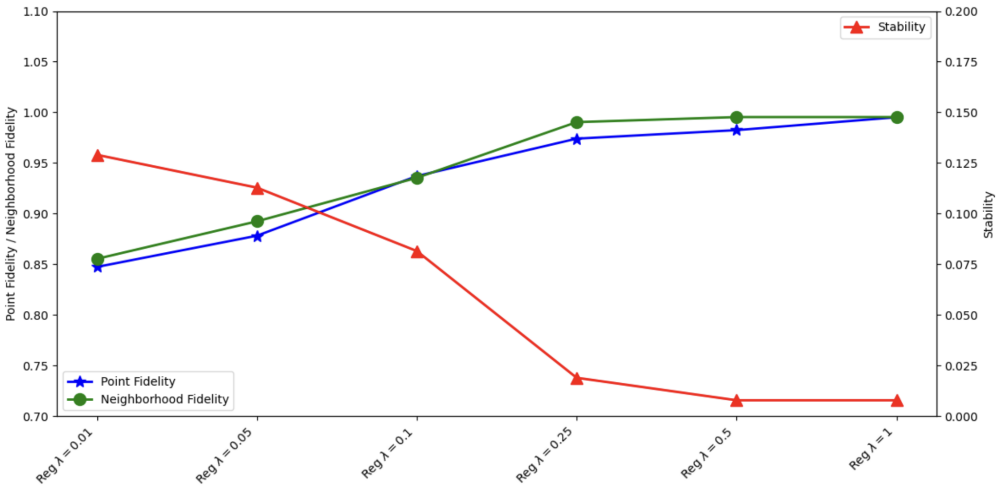



Figure 4.10: ML-100k Dataset results. Comparison of the Pre-hoc Framework with $k = 10$, for $\lambda = \{0.01, 0.05, 0.1, 0.25, 0.5, 1\}$ in point fidelity, neighborhood fidelity, and stability results. "Reg" means that regularization was used.

TABLE 4.4

ML-100k Dataset results. Comparison of the Pre-hoc Framework with $k = 10$, for $\lambda = \{0.01, 0.05, 0.1, 0.25, 0.5, 1\}$ in point fidelity, neighborhood fidelity, and stability results. “Reg" means that regularization was used.

| Explanation Method | Point Fidelity ↑ | Neighborhood Fidelity ↑ | Stability ↓ |
|---|---|---|---|
| No-regularization | $0.8183 \pm 0.3524$ | $0.8050 \pm 0.1268$ | $0.3524 \pm 0.0175$ |
| Reg $\lambda = 0.01$ | $0.8473 \pm 0.0351$ | $0.8553 \pm 0.1158$ | $0.1290 \pm 0.0010$ |
| Reg $\lambda = 0.05$ | $0.8781 \pm 0.0195$ | $0.8923 \pm 0.1043$ | $0.1128 \pm 0.0009$ |
| Reg $\lambda = 0.1$ | $0.9370 \pm 0.0230$ | $0.9353 \pm 0.0737$ | $0.0815 \pm 0.0019$ |
| Reg $\lambda = 0.25$ | $0.9740 \pm 0.0237$ | $0.9903 \pm 0.0329$ | $0.0189 \pm 0.0041$ |
| Reg $\lambda = 0.5$ | $0.9824 \pm 0.0234$ | $0.9953 \pm 0.0215$ | $0.0078 \pm 0.0010$ |
| Reg $\lambda = 1$ | $0.9951 \pm 0.0117$ | $0.9953 \pm 0.0215$ | $0.0078 \pm 0.0010$ |

white-box explainer are trained simultaneously. The joint optimization allows for a stronger alignment between the model's predictions and the explanations, leading to better local explainability.

In summary, the analysis of point fidelity, neighborhood fidelity, and stability metrics on the HELOC, Grasping, and ML-100k datasets highlights our proposed frameworks' more faithful local explainability compared to LIME. The explanations from our framework have the highest average neighborhood fidelity and the lowest total variation, indicating a high

level of agreement with our model's predictions and good stability across different neighborhoods. However, LIME explanations have moderate neighborhood fidelity with higher total variation, suggesting more variability in fidelity scores across different neighborhoods. Overall, the results indicate that the white-box explanations, especially those aligned with ours, provide more faithful and stable explanations in the local neighborhoods than LIME explanations.

**RQ 4.5 - Neighborhood Size: How does neighborhood size affect neighborhood fidelity, stability, and computational cost in Phase 2?**

As the neighborhood size (k) increases from 3 to 100 in Table 4.5 and Figure 4.11, the neighborhood fidelity increases. This suggests that larger neighborhoods better capture local patterns and provide more accurate explanations. The stability measure decreases as the neighborhood size increases. A lower stability value indicates more consistent explanations across different instances within the neighborhood. Larger neighborhoods provide more stable explanations. Finally, the computation time slightly increases with the neighborhood size, but the difference is insignificant. This implies that the computational cost is not impacted by the choice of neighborhood size, at least within the range of values considered.

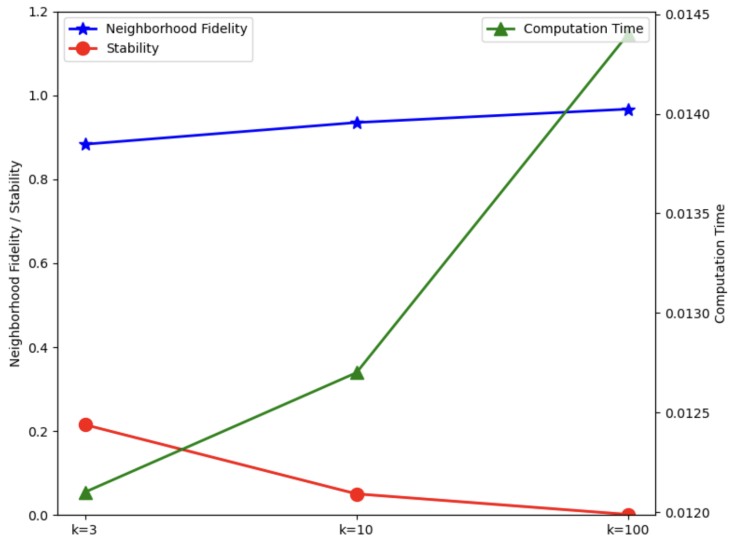



Figure 4.11: HELOC Dataset results on Pre-hoc for varying neighborhood size based on neighborhood fidelity, stability, and computation cost results. $\lambda = 0.25$.

**RQ 4.6 - Computational Efficiency**: What is the computational cost of generating explanations for our proposed frameworks compared to the LIME post-hoc method in Phase 2?

TABLE 4.5

HELOC Dataset results on Pre-hoc for varying neighborhood size based on neighborhood fidelity, stability, and computation cost results. $\lambda = 0.25$.

| Neighborhood Size | Neighborhood Fidelity ↑ | Stability ↓ | Computation Time |
|---|---|---|---|
| $k = 3$ | 0.8833 ± 0.1939 | 0.2152 ± 0.0175 | 0.0121 ± 0.0014 |
| $k = 10$ | 0.9350 ± 0.0381 | 0.0505 ± 0.0098 | 0.0127 ± 0.0009 |
| $k = 100$ | 0.9670 ± 0.0013 | 0.0015 ± 0.00006 | 0.0144 ± 0.0061 |

Computational efficiency is an essential factor to consider when evaluating the practicality and scalability of explanation methods. This research question compares the computational cost of generating explanations using our proposed frameworks and the state-of-the-art post-hoc method, LIME. All experiments were executed on an Intel(R) Xeon(R) CPU, 2.20GHz, and 13 GB RAM. An NVIDIA Tesla V100 GPU of 16 GB for CPU and GPU comparison is used. The results aim to provide insights into the computational trade-offs between Pre-hoc and post-hoc approaches and highlight the efficiency advantages of our proposed method. We assess the efficiency of both methods in terms of training time, the average time to explain a single instance, the total time to explain a single instance, and the total time for explaining the entire test set. Table 4.6 presents the calculation time comparison to generate explanations on the HELOC dataset, which consists of 988 instances in the test set. Our proposed Pre-hoc and Co-hoc methods include an additional training phase due to the local explainability regularization that takes 5.10 and 5.40 seconds for 7896 instances. This training time is a one-time cost incurred before generating explanations for individual instances. LIME, being a post-hoc method, does not have an additional phase during the training of the black-box model.

TABLE 4.6

HELOC Dataset. Computation time comparison for generating explanations on a test dataset of 100 instances. The average time is the computation time to explain a single instance.

| **Method** | **Additional Training Time (s)** | **Avg Explanation Time (s)** | **Total Time for Single Instance (s)** | **Total Time for Test Set(s)** |
|---|---|---|---|---|
| LIME | - | 0.3812 ± 0.0828 | 0.3812 ± 0.0828 | 376.625 |
| Pre-hoc | 5.1020 ± 0.0315 | 0.0110 ± 0.0015 | 5.1130 ± 0.0330 | 15.9700 |
| Co-hoc | 5.3960 ± 0.0330 | 0.0135 ± 0.0030 | 5.4095 ± 0.0360 | 18.7340 |

LIME takes an average of 0.38 seconds with a standard deviation of 0.082 seconds to generate an explanation for a single instance. On the other hand, our proposed Pre-hoc and Co-hoc frameworks take, respectively, an average of 0.011 and 0.013 seconds with a standard deviation of 0.0015 and 0.0030 seconds to generate an explanation for a single instance. This highlights the efficiency of our method, which is significantly faster than LIME in explaining individual instances. For our frameworks, the total time to explain a single instance includes both the training and the average explanation times. Therefore, the total time is 10.868 and 13.338 seconds.

LIME, being a post-hoc method, does not have a separate training phase, so the total time for explaining a single instance is the same as the average explanation time, which is 0.3812 seconds. When considering the total computation time for generating explanations for all 988 instances in the test dataset, LIME takes 376.625 seconds (approximately 6 minutes). The total computation time is calculated by multiplying the average explanation time by the number of instances (0.3812 seconds $\times$ 988 instances). Our proposed frameworks take approximately 11 and 13 seconds, respectively, to generate explanations for the entire test dataset. This total time includes the training time.

Overall, Table 4.6 highlights the trade-off between the one-time training cost of our proposed method and the efficiency gained in explaining individual instances. Although our method has a mandatory initial training time, it significantly outperforms LIME regarding the average time required to explain a single instance. This efficiency advantage becomes more apparent when generating explanations for many instances, as is the total time required to explain the entire test dataset. The results demonstrate that our proposed method has a balance between the initial training cost and the ability to generate explanations more efficiently compared to LIME. The faster explanation time per instance and the lower total time for the test set make our method more scalable and suitable, especially for scenarios where real-time or large-scale explanations are required.

Table 4.7 compares computation times for generating explanations on a test dataset of 100 instances using different methods on the Grasping dataset. The table provides insights into the computational efficiency of LIME and our proposed Pre-hoc and Co-hoc frameworks.

LIME does not require a separate training phase, as **it generates explanations for each instance independently**. The average time for LIME to explain a single instance is 0.3979 seconds, with a standard deviation of 0.0894 seconds. The total time to generate

TABLE 4.7

Grasping Dataset. Computation time comparison for generating explanations on a test dataset of 100 instances. The average (Avg) time is the computation time to explain a single instance. Results are based on five runs.

| Method | Additional Training Time (s) | Average Time (s) | Total Time for Single Instance (s) | Total Time for Test Set(s) |
|---|---|---|---|---|
| LIME | - | 0.3979 ± 0.0894 | 0.6114 ± 0.0752 | 56.9913 |
| Pre-hoc | 18.2300 | 0.1489 ± 0.0340 | 18.3789 ± 0.0340 | 33.1200 |
| Co-hoc | 16.7030 | 0.1580 ± 0.0445 | 16.8610 ± 0.0445 | 32.5030 |

explanations for the entire test set of 100 instances using LIME is 56.9913 seconds. On the other hand, our proposed Pre-hoc framework includes a training phase, which takes 18.230 seconds. The average time for generating an explanation for a single instance using the Pre-hoc framework is 0.1489 seconds, with a standard deviation of 0.0340 seconds. The total time for generating explanations for the 100 instances in the test set using the Pre-hoc framework is 33.1200 seconds, which includes the training time. The Co-hoc framework also consists of a training phase, which takes 16.7030 seconds. The average time for generating an explanation for a single instance using the Co-hoc framework is 0.1580 seconds, with a standard deviation of 0.0445 seconds. The total time for generating explanations for the 100 instances in the test set using the Co-hoc framework is 32.5030 seconds, including the training time.

Comparing the total time for generating explanations for the 100 instances in the test set, the Pre-hoc framework is the most efficient. Despite not requiring a separate training phase, LIME takes significantly longer, with a total time of 56.9913 seconds for the test set. These results demonstrate statistically significant computational efficiency (p-value $< 0.05$) of our proposed frameworks compared to LIME in both single instances and in total time. Although LIME generates explanations independently for each instance, our frameworks incorporate the explanation generation process into the training phase, resulting in faster explanation times during inference.

Table 4.8 shows a comparison of computation times for generating explanations on a test dataset of 100 instances using different hardware configurations (GPU and CPU) and datasets (ML100k and HELOC). In Single Epoch Training Time for the ML100k dataset, the GPU version takes an average of 0.9126 seconds per epoch, while the CPU version takes 0.9989 seconds. The difference in training time per epoch between the GPU and CPU is relatively small, with the GPU being slightly faster. For the HELOC dataset, the GPU

version takes an average of 0.3873 seconds per epoch, while the CPU version takes 0.1476 seconds. In this case, the CPU version is faster than the GPU version for training.

TABLE 4.8: Computation Cost comparison using CPU and GPU in Pre-hoc Explainability Framework.

| Dataset | Device | **Single Epoch Training Time (s)** | **Avg Single Explaination Time (s)** |
|---|---|---|---|
| ML-100k | CPU | 0.9989 ± 0.0661 | 0.0186 ± 0.0013 |
| | GPU | 0.9126 ± 0.4788 | 0.0146 ± 0.0014 |
| HELOC | CPU | 0.1476 ± 0.0021 | 0.0120 ± 0.0011 |
| | GPU | 0.3873 ± 0.4914 | 0.0149 ± 0.0034 |

In Average Single Explanation Time, for the ML100k dataset, the GPU version takes an average of 0.0146 seconds to generate a single explanation, while the CPU version takes 0.0186 seconds. The GPU version generates individual explanations slightly faster. For the HELOC dataset, the GPU version takes an average of 0.0149 seconds to generate a single explanation, while the CPU version takes 0.0120 seconds. In this case, the CPU version generates individual explanations faster.

For the ML100k and HELOC datasets, the total time for generating an explanation for a single instance is 30.1989 seconds on the GPU and 19.6094 seconds on the CPU. The CPU version appears to be faster regarding the total time for a single instance. For the ML100k and HELOC datasets, the total time for generating explanations for 100 instances is 6.2745 seconds on the GPU and 18.8441 seconds on the CPU. In this case, the GPU version is significantly faster than the CPU version.

GPUs are designed for parallel processing and are well-suited for tasks that can be parallelized, such as matrix operations and deep learning. They have many cores that can perform computations simultaneously, making them faster than CPUs for specific workloads. CPUs, on the other hand, are designed for general-purpose computing and sequential processing. They have fewer cores than GPUs but are more versatile and can handle a wide range of tasks efficiently.

It is important to note that comparing CPU and GPU performance may not be entirely meaningful when the code is not optimized for parallelization. Because explanations are generated sequentially without leveraging the GPU's parallel processing capabilities, the GPU does not provide significant speed improvements over the CPU.

The results in Table 4.8 show mixed performance between the GPU and CPU versions, with the CPU being faster in some cases (e.g., single epoch training time for HELOC, average single explanation time for HELOC) and the GPU being faster in others (e.g., total time for 100 instances). However, without parallelization in the code, the comparison does not reflect the true potential of GPU acceleration.

The implementation must be optimized for parallelization to fully utilize the GPU's capabilities and achieve significant speed improvements. This would involve identifying portions of the code that can be executed in parallel and leveraging GPU-specific libraries and frameworks to efficiently distribute the workload across the GPU cores. By parallelizing the explanation generation process and effectively utilizing the GPU, the computation cost could decrease significantly compared to running on the CPU.

Additionally, LIME, a popular post-hoc explainability method, does not support GPU usage due to it's dependency on scikit-learn library and only runs on the CPU. This limitation of LIME highlights the advantage of our proposed framework, which can leverage the power of GPUs to accelerate the explanation generation process, provided that the code is appropriately parallelized. Although the table compares computation times between GPU and CPU versions, without parallelization in the implementation, the comparison may not be expressive, and the actual benefits of GPU acceleration may not be realized. To fully take advantage of the power of GPUs, the code would need to be optimized for parallel processing. Additionally, the fact that LIME only supports CPU usage emphasizes the potential advantage of our proposed framework in terms of computational efficiency when properly utilizing GPUs.

### 4.5.7 Examples of Explanations

In this section, we present explanation examples of our explainer model, which provides both local and global explanations for the predictions made by the logistic regression model. Local explanations focus on understanding the importance of each feature for a specific test instance, allowing us to interpret how the model arrived at its prediction for that particular case. On the other hand, global explanations offer insights into the overall behavior of the model across all instances, revealing the features that generally have the most significant impact on the model's predictions. By examining both local and global feature importance scores, we gain a comprehensive understanding of the factors driving the

model's decisions at both the individual instance level and the broader pattern level.

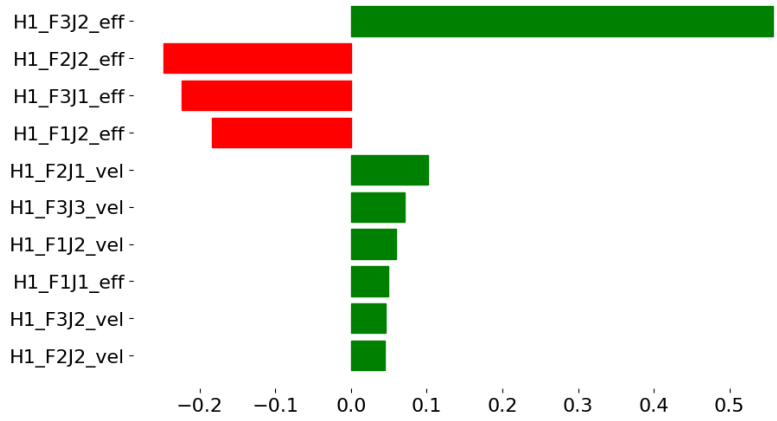



Figure 4.12: Grasping Data Set Explanation Example: Top 10 Feature Importance scores from the global explanation of the Pre-hoc framework, showing that increased effort exerted in Joint 2 of Finger 3 contributes tremendously to producing grasp failure, which is in contrast to increased efforts exerted at joint 2 of Fingers 1 and 2 and joint 1 of Finger 3, that lead to reducing grasp failure.

Global explainability addresses the need to broadly understand a machine learning model's decision-making, containing the entire model rather than individual predictions. The top 10 features of the Pre-hoc framework, as shown in Figure 4.12, indicate the relative importance of each feature and its contribution to predicting grasp robustness. Figure 4.12 shows that the most influential feature is H1F3J2eff, joint two effort on finger 3, which increases the model score by 0.5. It shows a strong positive correlation, suggesting a significant impact on grasp stability. In contrast, H1F2J2eff joint two effort in finger two negatively influences robustness by decreasing the model score by 0.25. Also, effort (e.g., torque) consistently has more effect on grasping results than the grasping velocity. This disparity in feature impact highlights the complex interaction between joint effort and velocity in determining the successful execution of a grasp. The analysis of these top features provides insights into the decision of the grasping process and reinforces the value of explainable AI in enhancing our understanding of machine learning models.

Local explainability focuses on understanding the reasoning behind a machine learning model's prediction for a single instance. Figure 4.13 illustrates the local feature importance scores for the Grasping dataset, providing insights into the instance-specific impact of each feature on the model's predictions.

In comparison to the global explanation, where effort (eff) features had a more pro-

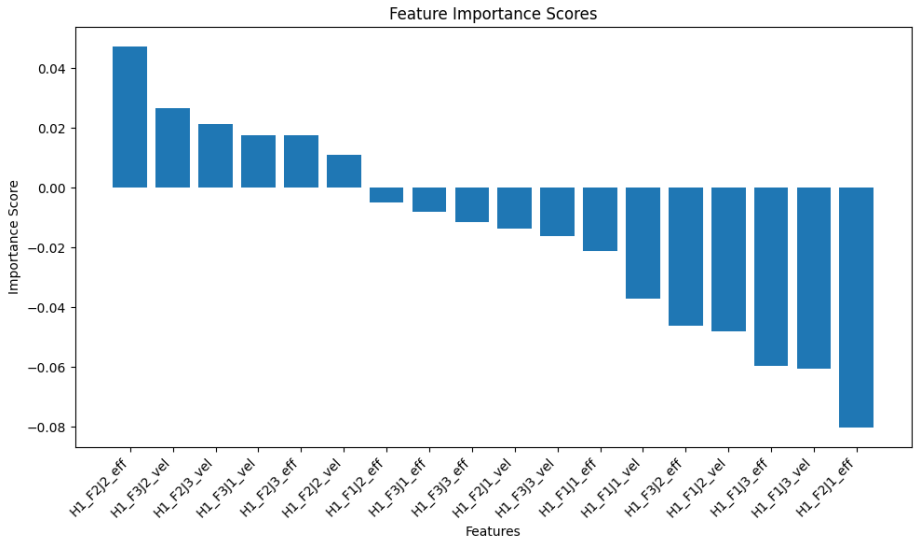



Figure 4.13: Grasping Dataset Local Explanation Example for Test Instance 143. The true label is predicted correctly by the predictor model in the Pre-hoc Framework, and the plotted Top10 feature importance scores are generated by the explainer model in the Pre-hoc Framework.

nounced effect than velocity (vel) features, the local explanation shows a more mixed impact from both types of features while effort may generally indicate grasp success in the broader model, velocity features can also be crucial in specific instances.

The differences between the local and global explanations underscore the variability in a model's decision-making process from one prediction to another. While global explanations provide an overview of the model's general behavior, local explanations reveal the nuances that can occur on a case-by-case basis, essential for understanding and trusting AI decisions in specific contexts.

Figure 4.14 illustrates the global feature importance scores for the HELOC dataset, providing insights into the overall impact of each feature on the model's predictions. The most influential feature is MaxDelq2PublicRecLast12M, which measures the maximum delinquency on public records in the last 12 months. This feature negatively impacts credit scores, suggesting that higher delinquency values significantly decrease the likelihood of getting a loan. Similarly, NumTrades90Ever2DerogPubRec, which represents the number of trades with derogatory public records, shows a substantial negative influence on the model's predictions. This implies that having more trades with derogatory records decreases the probability of the target variable. The global explanation also reveals that features related to credit inquiries and satisfactory trades play a notable role in the model's decision-making

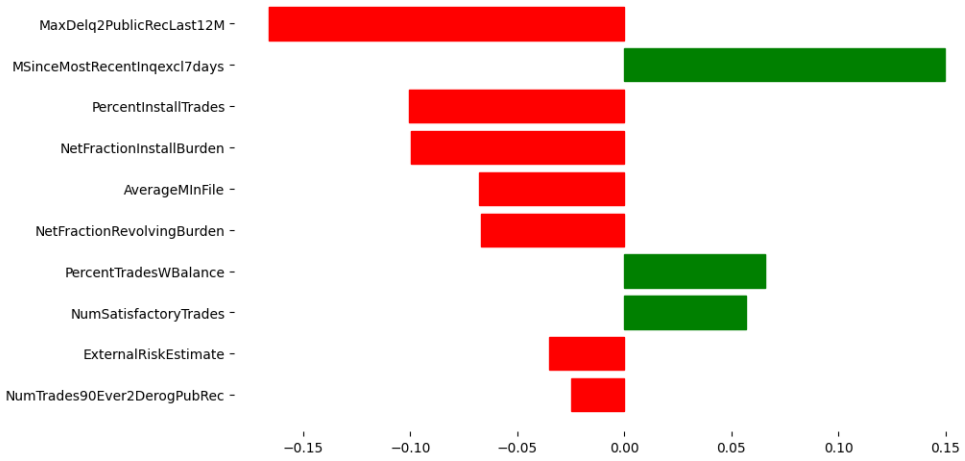



Figure 4.14: HELOC Dataset Global Explanation Example

process. MSinceMostRecentInqexcl7days, indicating the time since the most recent credit inquiry, has a positive impact on the predictions, while NumSatisfactoryTrades, which represents the number of satisfactory trades, exhibits a negative influence. This suggests that recent credit inquiries and fewer satisfactory trades are associated with a higher likelihood of the target outcome. Analyzing these top features provides valuable insights into the key drivers of the model's predictions. It enhances our understanding of the factors influencing the target outcome in the HELOC dataset.

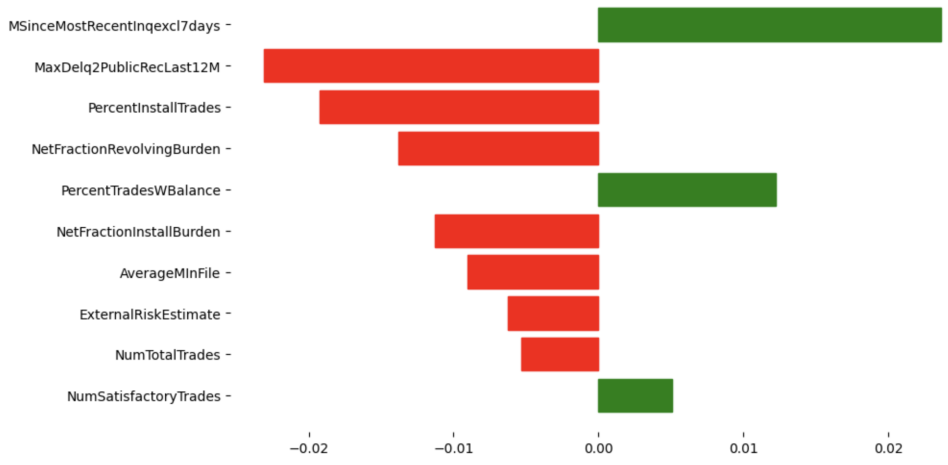



Figure 4.15: HELOC Dataset Local Explanation Example for Test Instance 1. The true label is predicted correctly by the predictor model in the Pre-hoc Framework, and the plotted Top10 feature importance scores are generated by the explainer model in the Pre-hoc Framework.

Figure 4.15 presents the local feature importance scores for test instance 1 in the

HELOC dataset, allowing us to understand the specific factors that influenced the model's prediction for this particular case. The most influential feature for this instance is MSinceMostRecentInqexcl7days, which has a strong positive impact, indicating that a more extended time since the most recent credit inquiry increases the likelihood of the target outcome for this instance. PercentInstallTrades and MaxDelq2PublicRecLast12M also show notable positive influences. However, NetFractionRevolvingBurden and NetFractionInstallBurden have negative impacts, suggesting that higher revolving and installment credit burdens decrease the probability of the target for this case.

Comparing the local explanation for test instance 1, Figure 4.15 to the global explanation Figure 4.14, we observe some similarities and differences. Both explanations highlight the importance of MaxDelq2PublicRecLast12M, although its directionality differs between the global and local views. MSinceMostRecentInqexcl7days also appears as a top feature in both explanations, with a consistent positive influence. However, the local explanation for test instance one emphasizes features like PercentInstallTrades and NetFractionRevolvingBurden, which are not as prominent in the global explanation. This suggests that while the global explanation provides an overall view of feature importance, the local explanation captures the unique characteristics and specific factors that influence the prediction for this particular instance.

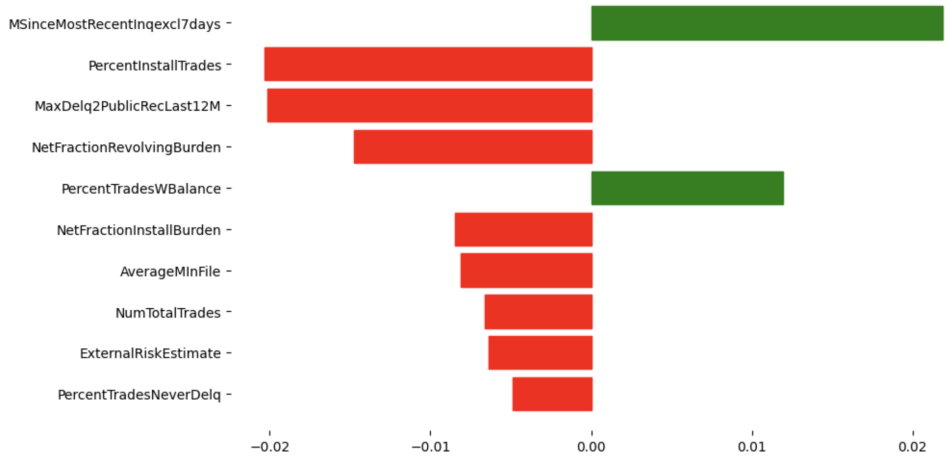



Figure 4.16: HELOC Dataset Local Explanation Example for Test Instance 50. The true label is **not predicted correctly** by predictor model in Pre-hoc Framework and the plotted Top10 feature importance scores generated by explainer model in Pre-hoc Framework.

Figure 4.16 presents the local feature importance scores for test instance 50 in the HELOC dataset. This allows us to understand the factors that influenced the model's pre-

diction for this case. The most influential feature for this instance is MSinceMostRecentInqexcl7days, exhibiting a strong positive impact. PercentInstallTrades and MaxDelq2PublicRecLast12M also show positive influences, although to a lesser extent than test instance 1. NetFractionRevolvingBurden and PercentTradesWBalance have negative impacts, indicating that a higher revolving credit burden and a higher percentage of trades with balance decrease the likelihood of the target outcome for this instance.

Comparing the local explanation for test instance 50, Figure 4.16 to the global explanation Figure 4.14 reveals some similarities, particularly in the importance of MSinceMostRecentInqexcl7days and MaxDelq2PublicRecLast12M. However, the local explanation for this instance emphasizes NetFractionRevolvingBurden and PercentTradesWBalance, which are not as prominent in the global explanation. Additionally, the directionality of some features, such as NetFractionInstallBurden, differs between the local and global views. These differences highlight the unique factors influencing the prediction for test instance 50 and underscore the importance of examining local explanations to better understand the model's decision-making process for specific cases.

Overall, the analysis of these local explanations compared to the global explanation demonstrates the value of considering both global and local perspectives when interpreting black-box models. While the global explanation provides an overall understanding of feature importance, the local explanations offer insights into the specific factors driving the model's predictions for individual instances, allowing for a more comprehensive and personalized interpretation of the model's behavior.

### 4.5.8 Summary

This chapter focused on optimizing our explainable machine learning frameworks for local explainability. We introduced the concept of local explainability, which aims to provide explanations that capture the model's behavior in the local neighborhood of a specific instance. We proposed a novel approach that leverages the Jensen-Shannon (JS) divergence and neighborhood information to generate instance-specific explanations.

We presented the problem formulation and discussed the optimization of local explainability using neighboring instances. We modified the objective and loss functions to incorporate the JS divergence and capture local variations in the predictions. We conducted a comparative analysis of our proposed frameworks with LIME, a popular post-hoc explain-

ability method. We evaluated the methods using metrics, including point fidelity, neighborhood fidelity, stability, and computational efficiency. The experimental results demonstrated the better performance of our Pre-hoc and Co-hoc frameworks in terms of local explainability, stability, and computational efficiency compared to LIME. We also investigated the impact of the explainability regularization parameter $\lambda$ on the local explainability of our frameworks. The results showed that increasing the value of $\lambda$ leads to improved point fidelity, neighborhood fidelity, and stability, highlighting the effectiveness of our regularization approach. Furthermore, we analyzed the computational efficiency of our frameworks and compared them with LIME. The results demonstrated that our frameworks, particularly the Pre-hoc framework, exhibit faster explanation generation times than LIME.

# CHAPTER 5

# CONCLUSION

In this dissertation, we contributed to the field of explainable artificial intelligence (XAI) by proposing novel approaches to enhance the explainability of black-box machine learning models. Chapter 1 introduced the motivation behind our research, highlighting the importance of explainability in AI systems and the challenges associated with interpreting complex models. In Chapter 2, we provided a comprehensive literature review covering various aspects of XAI, including inherently interpretable models, post-hoc explainability techniques, model-specific explainability approaches, and the different types of explanations. Chapter 3 presented our proposed frameworks, pre-hoc explainability, and co-hoc explainability, which integrate interpretability directly into the training process of black-box models. In Chapter 4, we further extended our pre-hoc framework to provide local explanations. Then we compared our approaches with the post-hoc explainability technique LIME regarding explanation quality, stability, and computational cost. In the following, we summarize our contributions, discuss the limitations of our work, and outline potential future research directions.

## 5.1 Summary of Contributions

We presented novel approaches to enhance the explainability of black-box machine learning models. Our main contributions are as follows:

1. We proposed two novel frameworks, pre-hoc explainability, and co-hoc explainability, which integrate interpretability directly into the training process of black-box models. These frameworks leverage the insights provided by an inherently interpretable white-box model to guide the learning of the black-box model, ensuring faithful and consistent explanations without requiring additional post-hoc computations. Our experiments confirmed the role of the explainability regularization coefficient $\lambda$ which allows users to control the trade-off between accuracy and explainability. Higher val-

ues of $\lambda$ lead to improved fidelity between the predictor and the explainer, enhancing the interpretability of the model, while maintaining a good level of accuracy. Hence the proposed frameworks provide users with the flexibility to adjust the model based on their specific needs and requirements.

2. We extended our pre-hoc explainability framework to provide local explanations by incorporating the Jensen-Shannon divergence as a regularization term. This extension allows our method to generate instance-specific explanations that capture the local behavior of the black-box model, similar to post-hoc methods like LIME. However, our approach integrates local explainability directly into the training process, ensuring that the explanations are faithful to the model's behavior and consistent with the model's predictions for similar instances.

3. Enhancing the accuracy of white-box models through the co-hoc learning framework. The white-box model, which is learned for the purpose of explaining the black-box predictor, achieves significantly higher prediction accuracy after the co-hoc learning process. This finding highlights the potential of the co-hoc in-training approach to improve the performance of white-box models, which are essential and required in certain high-risk and regulated application tasks in healthcare and legal decision-making.

4. We demonstrated the effectiveness of our approaches on two real-world benchmark datasets (one for credit risk assessment and the other for movie recommendation) and one simulated data set from the field of robotics, showing that our methods outperform traditional black-box models in terms of fidelity, while maintaining comparable accuracy. We also compared our methods with the LIME post-hoc explainability technique in terms of the quality and stability of the generated explanations and the computational cost associated with generating explanations. Our results indicate that our approaches provide more faithful and consistent explanations across different instances, as measured by the metrics of global fidelity, local fidelity, and stability. Furthermore, our methods exhibit significantly lower computational costs than LIME, as they do not require additional post-hoc computations for generating explanations.

5. We provided a theoretical analysis of our approaches, showing that they can be seen as a form of regularized learning that balances the trade-off between accuracy and

interpretability. We proved that incorporating the interpretable white-box model and the JS divergence as regularization terms in the loss function encourages the black-box model to learn a decision boundary that is more aligned with the interpretable model, leading to improved global and local explainability, while striking a better balance between accuracy and interpretability compared to post-hoc methods like LIME.

## 5.2 Limitations and Future Work

Despite the promising results and contributions of our work, there are several limitations and avenues for future research:

- Model selection: In our current implementation, we focused on using sparse linear models as the inherently interpretable white-box models. While sparse linear models balance interpretability and predictive power, they may not always be the most suitable choice for every dataset or application. Future work could explore integrating other interpretable models, such as decision trees or rule-based systems, into our frameworks to provide more diverse and domain-specific explanations.

- Explanation types: Our current frameworks focus primarily on feature-based explanations, highlighting the importance of individual features in the model's decision-making process. While feature-based explanations are widely used and practical, they may not always be the most suitable explanation for every application or user. Future research could explore integrating other explanation types, such as rule-based, concept-based, or example-based explanations, into our frameworks. Rule-based explanations provide human-readable IF-THEN rules that capture the model's decision logic, making them easily interpretable for domain experts. Concept-based explanations align the model's behavior with human-understandable concepts, bridging the gap between the model's internal representation and the user's domain knowledge. Example-based explanations present similar instances from the training data to justify the model's predictions, providing a more intuitive understanding of the model's reasoning. By incorporating a diverse range of explanation types, we can cater to the specific needs and preferences of different users and application domains, enhancing the interpretability and usefulness of the generated explanations.

- Scalability: Although our approaches have shown significant computational advantages *at testing time* over post-hoc methods like LIME, the scalability of our methods to huge datasets and complex model architectures remains to be investigated. Future research could focus on developing more efficient optimization techniques and parallelization strategies to enhance the scalability of our frameworks.

- Evaluation metrics: We used evaluation metrics, such as global fidelity, local fidelity, and stability, to assess the quality of the generated explanations. However, there is still a lack of consensus in the XAI community regarding the most appropriate metrics for evaluating explainability methods. Future work could involve collaborating with domain experts and end-users to develop more comprehensive and application-specific evaluation frameworks that capture the usefulness and interpretability of explanations from a human perspective.

- Human-in-the-loop explanations: While our approaches generate more faithful and consistent explanations compared to post-hoc methods, the ultimate goal of explainable AI is to provide meaningful and actionable insights for human users. Future research could explore the integration of human feedback and domain knowledge into the explanation generation process, allowing for more interactive and user-centric explanations.

- Fairness and bias: Explainable AI techniques have the potential to uncover and mitigate biases in machine learning models. However, our current frameworks do not explicitly address the issues of fairness and bias. Future work could investigate how our approaches can be extended to detect and mitigate biases, ensuring that the generated explanations are faithful, consistent but also fair, and unbiased.

- Data Quantity and Quality: The transparency of the white-box model may depend on the quality and quantity of the training data, as well as the complexity and heterogeneity of the underlying distribution. In particular, if the data are noisy or biased, or if the true relationship between the input and output variables is highly nonlinear or ambiguous, the white-box explainer model in our proposed framework could be easily replaced by alternative differentiable white-box models, such as rule-based or sparse additive models.

- Real-world applications: While we have demonstrated the effectiveness of our approaches on benchmark datasets, future research could focus on applying our frameworks to real-world applications in domains such as healthcare, finance, and criminal justice. Collaborating with domain experts and stakeholders in these fields could provide valuable insights into the practical challenges and requirements for deploying explainable AI solutions in real-world settings.

## 5.3 Conclusion

In conclusion, this work contributed to the field of explainable AI by proposing novel frameworks that integrate interpretability directly into the training process of black-box models. Our approaches, pre-hoc and co-hoc explainability, provide faithful, consistent, and computationally efficient explanations that outperform post-hoc methods like LIME. By extending our frameworks to provide local explanations and conducting theoretical analyses, we have demonstrated the effectiveness and versatility of our frameworks in enhancing the explainability of black-box models.

However, explainable AI remains an active and challenging research area, with many open questions and opportunities for future work. This dissertation's limitations and future directions highlight the need for continued research efforts to develop more scalable, user-centric, and application-specific explainability techniques. By addressing these challenges and collaborating with domain experts and end-users, we can move closer to building transparent, accountable, and trustworthy AI systems that benefit society as a whole.

## REFERENCES


[1] D. Gunning, "Explainable artificial intelligence (xai)," 2017.

[2] David Alvarez-Melis and Tommi S. Jaakkola, "On the robustness of interpretability methods," 2018.

[3] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim, "Sanity checks for saliency maps," 2020.

[4] Amirata Ghorbani, Abubakar Abid, and James Zou, "Interpretation of neural networks is fragile," 2018.

[5] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.

[6] Finale Doshi-Velez and Been Kim, "Towards a rigorous science of interpretable machine learning," 2017.

[7] 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.

[8] Finale Doshi-Velez and Been Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.

[9] Pang Wei Koh and Percy Liang, "Understanding black-box predictions via influence functions," in *International Conference on Machine Learning*, 2017, pp. 1885–1894.

[10] Sebastian Lapuschkin, Stephan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller, "Unmasking clever hans predictors and assessing what machines really learn," *Nature communications*, vol. 10, no. 1, pp. 1096, 2019.

[11] Bryce Goodman and Seth Flaxman, "European union regulations on algorithmic decision-making and a "right to explanation"," *AI magazine*, vol. 38, no. 3, pp. 50–57, 2017.

[12] Sandra Wachter, Brent Mittelstadt, and Luciano Floridi, "Why a right to explanation of automated decision-making does not exist in the general data protection regulation," *International Data Privacy Law*, vol. 7, no. 2, pp. 76–99, 2017.

[13] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel, "Fairness through awareness," in *Proceedings of the 3rd innovations in theoretical computer science conference*, 2012, pp. 214–226.

[14] Andrew D Selbst and Solon Barocas, "The intuitive appeal of explainable machines," *Fordham Law Review*, vol. 87, pp. 1085, 2018.

[15] Gagan Bansal, Tongshuang Wu, Joyce Zhou, Raymond Fok, Besmira Nushi, Ece Kamar, Marco Tulio Ribeiro, and Daniel Weld, "Updates in human-ai teams: Understanding and addressing the performance/compatibility tradeoff," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 2429–2437.

[16] Isaac Lage, Emily Chen, Jeffrey He, Menaka Narayanan, Been Kim, Sam Gershman, and Finale Doshi-Velez, "An evaluation of the human-interpretability of explanation," *arXiv preprint arXiv:1902.00006*, 2019.

[17] 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, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 1135–1144.

[18] Scott M Lundberg and Su-In Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., pp. 4765–4774. Curran Associates, Inc., 2017.

[19] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 336–359, oct 2019.

[20] Sebastian Bordt, Michèle Finck, Eric Raidl, and Ulrike von Luxburg, "Post-hoc explanations fail to achieve their purpose in adversarial contexts," in *2022 ACM Conference on Fairness, Accountability, and Transparency*. jun 2022, ACM.

[21] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju, "Fooling lime and shap: Adversarial attacks on post hoc explanation methods," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, New York, NY, USA, 2020, AIES '20, p. 180–186, Association for Computing Machinery.

[22] David Alvarez-Melis and Tommi S. Jaakkola, "Towards robust interpretability with self-explaining neural networks," *CoRR*, vol. abs/1806.07538, 2018.

[23] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim, "Towards automatic concept-based explanations," in *Advances in Neural Information Processing Systems*. 2019, vol. 32, Curran Associates, Inc.

[24] Cynthia Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," 2019.

[25] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki, "The dangers of post-hoc interpretability: Unjustified counterfactual explanations," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. 7 2019, pp. 2801–2807, International Joint Conferences on Artificial Intelligence Organization.

[26] Erik Englesson and Hossein Azizpour, "Generalized jensen-shannon divergence loss for learning with noisy labels," 2021.

[27] David Alvarez-Melis and Tommi S Jaakkola, "On the robustness of interpretability methods," in *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[28] John Neter, Michael H Kutner, Christopher J Nachtsheim, and William Wasserman, *Applied linear statistical models*, Irwin Chicago, 1996.

[29] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant, *Applied logistic regression*, John Wiley & Sons, 2013.

[30] P. McCullagh and J.A. Nelder, *Generalized Linear Models, Second Edition*, Chapman and Hall/CRC Monographs on Statistics and Applied Probability Series. Chapman & Hall, 1989.

[31] Gerhard Tutz, *Poisson Regression*, pp. 1075–1077, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

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

[33] Arthur E. Hoerl and Robert W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. pp. 55–67, 1970.

[34] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society (Series B)*, vol. 58, pp. 267–288, 1996.

[35] Hui Zou and Trevor Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 2, pp. 301–320, 2005.

[36] J. Ross Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[37] J. Ross Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[38] L. Breiman, Jerome H. Friedman, Richard A. Olshen, and C. J. Stone, "Classification and regression trees," *Biometrics*, vol. 40, pp. 874, 1984.

[39] S. Rasoul Safavian and David Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[40] Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone, *Classification and Regression Trees*, Wadsworth Brooks/Cole Advanced Books Software, Monterey, CA, 1984.

[41] Tin Kam Ho, "Random decision forests," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, 1995, vol. 1, pp. 278–282 vol.1.

[42] Jerome H. Friedman, "Greedy function approximation: A gradient boosting machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[43] Leo Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[44] Thibault Laugel, Marie-Jeanne Lesot, Christophe Marsala, Xavier Renard, and Marcin Detyniecki, "The dangers of post-hoc interpretability: Unjustified counterfactual explanations," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. 7 2019, pp. 2801–2807, International Joint Conferences on Artificial Intelligence Organization.

[45] Eric Tjoa and Cuntai Guan, "A survey on explainable artificial intelligence (xai): Toward medical xai," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[46] Damien Garreau and Ulrike Von Luxburg, "Explaining the explainer: A first theoretical analysis of lime," in *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, 2020.

[47] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, "Anchors: High-precision model-agnostic explanations," *AAAI Conference on Artificial Intelligence*, 2018.

[48] Scott M Lundberg and Su-In Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, 2017.

[49] Lloyd S Shapley, "A value for n-person games," *Contributions to the Theory of Games*, vol. 2, no. 28, pp. 307–317, 1953.

[50] Scott M Lundberg, Bala Nair, Monica S Vavilala, Mayumi Horibe, Michael J Eisses, Trevor Adams, David E Liston, Daniel K-W Low, Shu-Fang Newman, Jerry Kim, and Su-In Lee, "Explainable machine-learning predictions for the prevention of hypoxaemia during surgery," *Nature Biomedical Engineering*, vol. 2, no. 10, pp. 749–760, 2018.

[51] Christoph Molnar, *Interpretable Machine Learning*, Lulu.com, 2020.

[52] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee, "Consistent individualized feature attribution for tree ensembles," in *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 2018.

[53] 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.

[54] Gabriëlle Ras, Luca Ambrogioni, Pim Haselager, Marcel A. J. van Gerven, and Umut Güçlü, "Explainable 3d convolutional neural networks by learning temporal transformations," 2020.

[55] Kevin Fauvel, Tao Lin, Véronique Masson, Élisa Fromont, and Alexandre Termier, "Xcm: An explainable convolutional neural network for multivariate time series classification," 2020.

[56] Siqi Miao, Miaoyuan Liu, and Pan Li, "Interpretable and generalizable graph learning via stochastic attention mechanism," 2022.

[57] Robert Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[58] Arthur E Hoerl and Robert W Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[59] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," in *The journal of machine learning research*, 2014, vol. 15, pp. 1929–1958.

[60] Lutz Prechelt, "Early stopping-but when?," *Neural Networks: Tricks of the trade*, pp. 55–69, 1998.

[61] Gregory Plumb, Denali Molitor, and Ameet Talwalkar, "Model agnostic supervised local explanations," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2018, NIPS'18, p. 2520–2529, Curran Associates Inc.

[62] Mike Wu, Michael C. Hughes, Sonali Parbhoo, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez, "Beyond sparsity: Tree regularization of deep models for interpretability," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*. 2018, AAAI'18/IAAI'18/EAAI'18, AAAI Press.

[63] Zachary C. Lipton, "The mythos of model interpretability," 2017.

[64] Jianbo Chen, Le Song, Martin J. Wainwright, and Michael I. Jordan, "Learning to explain: An information-theoretic perspective on model interpretation," 2018.

[65] Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez, "Right for the right reasons: Training differentiable models by constraining their explanations," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 2662–2670.

[66] Guang-He Lee, David Alvarez-Melis, and Tommi S. Jaakkola, "Game-theoretic interpretability for temporal modeling," 2018.

[67] Guang-He Lee, Wengong Jin, David Alvarez-Melis, and T. Jaakkola, "Functional transparency for structured data: a game-theoretic approach," in *International Conference on Machine Learning*, 2019.

[68] Gregory Plumb, Maruan Al-Shedivat, Ángel Alexander Cabrera, Adam Perer, Eric Xing, and Ameet Talwalkar, "Regularizing black-box models for improved interpretability," in *Advances in Neural Information Processing Systems*. 2020, vol. 33, pp. 10526–10536, Curran Associates, Inc.

[69] Anirban Sarkar, Deepak Vijaykeerthy, Anindya Sarkar, and Vineeth N Balasubramanian, "A framework for learning ante-hoc explainable models via concepts," 2021.

[70] Alejandro Barredo Arrieta, Natalia D'ıaz-Rodr'ıguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garc'ıa, Sergio Gil-L'opez, Daniel Molina, Richard Benjamins, et al., "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information Fusion*, vol. 58, pp. 82–115, 2020.

[71] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi, "A survey of methods for explaining black box models," *ACM computing surveys (CSUR)*, vol. 51, no. 5, pp. 1–42, 2018.

[72] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)," in *International Conference on Machine Learning*, 2018, pp. 2668–2677.

[73] Arnaud Van Looveren and Janis Klaise, "Global aggregations of local explanations for black box models," in *ECML PKDD 2019 Workshop on Automating Data Science*, 2019.

[74] Sana Tonekaboni, Shalmali Joshi, Melissa D McCradden, and Anna Goldenberg, "What clinicians want: contextualizing explainable machine learning for clinical end use," *Machine learning for healthcare conference*, pp. 359–380, 2019.

[75] Niklas Bussmann, Paolo Giudici, Dimitri Marinelli, and Jochen Papenbrock, "Explainable ai in fintech risk management," *Frontiers in Artificial Intelligence*, vol. 3, pp. 26, 2020.

[76] Umang Bhatt, Alice Xiang, Shubham Sharma, Adrian Weller, Ankur Taly, Yunhan Jia, Joydeep Ghosh, Ruchir Puri, Jos'e MF Moura, and Peter Eckersley, "Explainable machine learning in deployment," in *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, 2020, pp. 648–657.

[77] Amina Adadi and Mohammed Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.

[78] Amit Das and Paul Rad, "Opportunities and challenges in explainable artificial intelligence (xai): A survey," *arXiv preprint arXiv:2006.11371*, 2020.

[79] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, David Madigan, et al., "Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model," *Annals of Applied Statistics*, vol. 9, no. 3, pp. 1350–1371, 2015.

[80] Himabindu Lakkaraju, Stephen H Bach, and Jure Leskovec, "Interpretable decision sets: A joint framework for description and prediction," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1675–1684.

[81] Christoph Molnar, *Interpretable Machine Learning*, 2019, `https://christophm.github.io/interpretable-ml-book/`.

[82] Ke Yang, Xavier Amatriain, Xiang Ren, Ziming Hong, and Yong Li, "Scalable and interpretable product recommendations via overlapping co-clustering," *arXiv preprint arXiv:1709.01987*, 2017.

[83] Himabindu Lakkaraju, Ece Kamar, Rich Caruana, and Jure Leskovec, "Interpretable explorable approximations of black box models," in *Proceedings of the 2017 ICML Workshop on Human Interpretability in Machine Learning*, 2017, pp. 34–42.

[84] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin, "Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation," in *Journal of Computational and Graphical Statistics*. Taylor Francis, 2015, vol. 24, pp. 44–65.

[85] Mukund Sundararajan and Amir Najmi, "Many shapley values," *arXiv preprint arXiv:2002.12296*, 2020.

[86] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim, "Towards automatic concept-based explanations," in *Advances in Neural Information Processing Systems*, 2019, vol. 32.

[87] Finale Doshi-Velez and Been Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.

[88] Amirata Ghorbani, Abubakar Abid, and James Zou, "Interpretation of neural networks is fragile," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 3681–3688.

[89] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang, "Concept bottleneck models," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5338–5348.

[90] Zhi Chen, Yijia Bei, and Cynthia Rudin, "Concept whitening for interpretable image recognition," in *Nature Machine Intelligence*. Nature Publishing Group, 2020, vol. 2, pp. 772–782.

[91] Jacob Bien and Robert Tibshirani, "Prototype selection for interpretable classification," in *The Annals of Applied Statistics*. JSTOR, 2011, pp. 2403–2424.

[92] Eric Wallace, Shi Feng, and Jordan Boyd-Graber, "Interpreting neural networks with nearest neighbors," in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2018, pp. 136–144.

[93] Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar, "Representer point selection for explaining deep neural networks," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[94] Sandra Wachter, Brent Mittelstadt, and Chris Russell, "Counterfactual explanations without opening the black box: Automated decisions and the gdpr," in *Harv. JL Tech.* HeinOnline, 2017, vol. 31, p. 841.

[95] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," in *arXiv preprint arXiv:1412.6572*, 2014.

[96] Scott M Lundberg, Gabriel G Erion, and Su-In Lee, "Local explanations for global machine learning models: Feature importance scores with shapley values," in *arXiv preprint arXiv:2005.04118*, 2020.

[97] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell, "Generating visual explanations," in *European Conference on Computer Vision*. Springer, 2016, pp. 3–19.

[98] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," in *arXiv preprint arXiv:1503.02531*, 2015.

[99] Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil, "Model compression," *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 535–541, 2006.

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

[101] Yoav Freund and Robert E Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[102] 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," *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1721–1730, 2015.

[103] Steffen Rendle, "Factorization machines," in *Proceedings of the 2010 IEEE International Conference on Data Mining*, USA, 2010, ICDM '10, p. 995–1000, IEEE Computer Society.

[104] Thorsten Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2002, KDD '02, p. 133–142, Association for Computing Machinery.

[105] Nataliya Sokolovska, Yann Chevaleyre, and Jean-Daniel Zucker, "A provable algorithm for learning interpretable scoring systems," in *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, Amos Storkey and Fernando Perez-Cruz, Eds., Playa Blanca, Lanzarote, Canary Islands, 09–11 Apr 2018, vol. 84 of *Proceedings of Machine Learning Research*, pp. 566–574, PMLR.

[106] Liang Lan and Yu Geng, "Accurate and interpretable factorization machines," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 4139–4146, Jul. 2019.

[107] Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Azzurra Ragone, and Joseph Trotta, "How to make latent factors interpretable by feeding factorization machines with knowledge graphs," in *Lecture Notes in Computer Science*, pp. 38–56. Springer International Publishing, 2019.

[108] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua, "Attentional factorization machines: Learning the weight of feature interactions via attention networks," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2017, IJCAI'17, p. 3119–3125, AAAI Press.

[109] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua, "Attentive collaborative filtering: Multimedia recommendation with item- and component-level attention," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, 2017, SIGIR '17, p. 335–344, Association for Computing Machinery.

[110] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua, "Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning," 2017.

[111] Frank Nielsen and Sylvain Boltz, "The burbea-rao and bhattacharyya centroids," *IEEE Transactions on Information Theory*, vol. 57, no. 8, pp. 5455–5466, 2011.

[112] Thomas M Cover and Joy A Thomas, "Information theory and statistics," *Elements of information theory*, vol. 1, no. 1, pp. 279–335, 1991.

[113] Frank Nielsen, "On the jensen–shannon symmetrization of distances relying on abstract means," *Entropy*, vol. 21, no. 5, pp. 485, may 2019.

[114] GroupLens, "Movielens 100k dataset," `https://grouplens.org/datasets/movielens/100k/`.

[115] FICO, "The fico heloc dataset," `https://community.fico.com/s/explainable-machine-learning-challenge?tabset-3158a=2`.

[116] W. James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu, "Definitions, methods, and applications in interpretable machine learning," *Proceedings of the National Academy of Sciences*, vol. 116, no. 44, pp. 22071–22080, oct 2019.

[117] Steffen Rendle, "Factorization machines," in *2010 IEEE International Conference on Data Mining*, 2010, pp. 995–1000.

[118] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," 2017.

[119] Shadow Robot, "Smart grasping sandbox," `https://github.com/shadow-robot/smart_grasping_sandbox`, 2023.

[120] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala, "Pytorch: An imperative style, high-performance deep learning library," 2019.

CURRICULUM VITAE

**Name**

Asuman Cagla Acun

**Education**

Ph.D., University of Louisville, Louisville, KY, April 2024

M.S. , University of Louisville, Louisville, KY, May 2017

B.S. Hacettepe University, Ankara, Turkiye, June 2013

**Professional Positions**

Graduate Research Assistant in Advanced Automation and Robotics Research Institute (LARRI),
University of Louisville,
September 2022 - April 2024

Graduate Student Assistant in Resources for Academic Achievement (REACH),
University of Louisville,
August 2018 - June 2021

Software Engineer in Innova IT Solutions, Ankara, Turkey,
June 2013 - August 2014

**Honors and Awards**

Best Student Paper Award, Oct 2023, Given by IEEE Technical Committee of Intelligent Informatics at IEEE WI-IAT'23 Conference

CSE Arthur M. Riehl Award, May 2021, Given by J. B. Speed School of Engineering, University of Louisville. The award is given to a graduate student with excellent academic performance and contributions to the department activities.

Best Poster Presentation Award, Feb 2019, Given by Graduate Student Regional Research Conference (GSRRC) Committee

**Publications**

1. C. Acun and O. Nasraoui, "In-training explainability frameworks: A method to make black-box machine learning models more explainable," in 2023 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), 2023, pp. 230–237. doi: 10.1109/WI-IAT59888.2023.00036. (*Winner of Best Student Paper Award*)

2. K. D. Spurlock, C. Acun, E. Saka, and O. Nasraoui, Chatgpt for conversational recommendation: Refining recommendations by reprompting with feedback, 2024. arXiv: 2401.03605 [cs.IR].

3. C. Acun and R. Acun, "Gai-enhanced assignment framework: A case study on generative ai powered history education," in NeurIPS Workshop on Generative AI for Education (GAIED), New Orleans, Louisiana: NeurIPS, Dec. 2023. [Online]. Available: https://gaied.org/neurips2023/files/48/48%5C_paper.pdf.

4. M. Boujelbene, K. Damak, A. C. Acun, et al., "A data-science approach to flagging non-retention in engineering enrollment data," in 2020 ASEE Virtual Annual Conference Content Access, Virtual Online: ASEE Conferences, Jun. 2020.[Online]. Available: https://peer.asee.org/33996.

5. A. C. Acun, O. Nasraoui, and J. L. Hieb, "Work in progress: Finding the right questions: Using data science to close the loop with classroom response systems," in 2019 ASEE Annual Conference & Exposition, Tampa, Florida: ASEE Conferences, Jun. 2019. [Online]. Available: https://peer.asee.org/33619.

6. A. C. Acun, J. L. Hieb, and O. Nasraoui, "Using a data science pipeline for course data: A case study analyzing, heterogeneous student data in two flipped classes," in 2019 ASEE Annual Conference & Exposition, Tampa, Florida: ASEE Conferences, Jun. 2019. [Online]. Available: https://peer.asee.org/33492.