TU WIEN Informatics

Recognition of Differences between two Binary Black Box Classifiers to create Explanations using Model-Agnostic Methods

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

066926 Business Informatics

by

Andreas Staufer, B.Sc.
Registration Number 00951825

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber

Vienna, 7th December, 2021

Andreas Staufer	Andreas Rauber

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Andreas Staufer, B.Sc.

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 7. Dezember 2021

Andreas Staufer

Danksagung

Mein besonderer Dank gilt meinem Betreuer Prof. Dr. Andreas Rauber, der es mir ermöglicht hat, diese Arbeit in der Information and Software Engineering Group (IFS) zu schreiben. Er hat mich von Anfang an bei meiner Arbeit großartig unterstützt. Viele kritische und fordernde Diskussionen haben geholfen die Arbeit fertigzustellen. Ich bin auch dankbar dafür, dass ich durch ihn sehr viel Neues lernen konnte und mein Wissen in Machine Learning und insbesondere in Explainable Machine Learning enorm erweitern konnte.

An dieser Stelle möchte ich ebenfalls allen Personen danken, die mich unterstützt haben. Hierbei möchte ich meinen Freunden Johannes, Stefan und Thomas danke sagen, die mir immer mit Rat beiseite standen.

Ebenfalls möchte ich meinem Arbeitgeber und Stefan Holzer danken. Stefan bot mir flexible Arbeitsbedingungen, die es mir ermöglicht haben, mein Studium neben der Arbeit abzuschließen.

Des Weiteren möchte ich meiner Freundin Bernadette danken, die sehr viel Geduld aufgebracht hat und mir den Rücken in vielen Situationen freigehalten hat.

Zuletzt möchte ich meiner Familie und insbesondere meinen Eltern danken. Ohne ihre bedingungslose Unterstützung wäre das Studium nicht möglich gewesen.

Acknowledgements

My special thanks go to my supervisor Prof. Dr. Andreas Rauber, who made it possible to write this thesis in the Information and Software Engineering Group (IFS). He has given me great support in my work right from the start. Many critical and demanding discussions helped to finish the thesis. I am also grateful that I could learn a lot of new things. Furthermore, I could enormously expand my knowledge in machine learning and especially in explainable machine learning.

At this point, I would also like to thank everyone who has supported me. I want to say thank you to my friends Johannes, Stefan, and Thomas, who were always at my side with good advice.

Furthermore, I would like to thank my employer and Stefan Holzer. Stefan offered me flexible working conditions to complete my studies alongside work.

I would also like to thank my girlfriend Bernadette, who was very patient and kept my back in many situations.

Finally, I would like to thank my family and especially my parents. Without their unconditional support, my Master's would not have been possible.

Kurzfassung

Der vermehrte Einsatz von Black Boxes als Entscheidungssysteme in wichtigen Bereichen unseres Lebens steht in der Kritik. Black Boxes besitzen die unerwünschte Eigenschaft, dass deren Entscheidungsgrundlage für einen Menschen nicht nachvollziehbar ist. Interpretierbare Resultate sind jedoch aus verschiedenen Gründen wie rechtlichen, ethischen und sicherheitstechnischen Aspekten notwendig. Daher wurden unterschiedliche Methoden entwickelt und vorgestellt, um Erklärungen für die Entscheidungen einer einzelnen Black Box zu liefern. Der LORE-Ansatz ist eine vielversprechende modell-agnostische Methode, um die Ergebnisse der Black Box für einen bestimmten Fall verständlich zu erklären. Modell-agnostische Methoden sind jedoch darauf ausgelegt, die Ergebnisse eines einzelnen Black Box-Modells zu interpretieren. Wir stellen DiRo2C vor, um die unterschiedlichen Entscheidungen zweier binärer Black Box Klassifizierer zu erklären.

Unser Ansatz verwendet einen modifizierten genetischen Algorithmus von LORE, um einen synthetischen ausgewogenen Datensatz generieren zu können. **DiRo2C** verwendet diesen generierten Datensatz, um einen Klassifizierer zu trainieren, der die lokalen Unterschiede nahe einer bestimmten Instanz zwischen den Black Boxen erkennt. Durch Auswahl verschieden positionierter Instanzen und Generierung von Datensätzen, kann ein globaler Explainer trainiert werden. Dazu wird ein erklärbarer, auf einem Entscheidungsbaum-basierenden Klassifizierer verwendet. Der Klassifizierer kann ebenfalls durch Anwendung eines beliebigen erklärbaren KI (Künstliche Intelligenz)-Ansatzes interpretiert werden. **DiRo2C** unterstützt das Training eines binären Klassifizierer, der unterschiedliche Ergebnisse zwischen den Black Boxen vorhersagt, und einen Multiklassen-Prädiktor, der jede mögliche Kombination der binären Black Box-Ergebnisse vorhersagt. Der modifizierte genetische Neighborhood Algorithmus wurde gegen andere Strategien getestet. Unsere Simulationen und Experimente zeigen, dass der binäre Klassifizierer, der durch unseren modifizierten genetischen Ansatz trainiert wird, andere implementierte Lösungen in Bezug auf Genauigkeit und Qualität der erkannten Unterschiede weit übertrifft.

Wir evaluieren die Leistung der Klassifizierer, die auf Basis der verschiedenen Datenansätze für drei verschiedene Datensätze trainiert werden, indem wir eine stratifizierte 10-fach-Kreuzvalidierung anwenden. Darüber hinaus verwenden wir Metriken wie *Accuracy*, F_1-*Score* und *Pearson Correlation Coefficient*. Wir manipulieren eine Black Box, indem wir ein bestimmtes Attribut aller Instanzen ändern, um Unterschiede zwischen den Black Boxen zu erzwingen. Die gefundenen Unterschiede werden ebenfalls auf Korrektheit überprüft und ob der Klassifikator die tatsächlichen Unterschiede erkennt.

Abstract

The increased use of black boxes as decision systems, especially in crucial areas of our lives, is under criticism. Black boxes have the undesirable characteristic that the basis for making decisions is incomprehensible for a human being. However, interpretable results are necessary for different reasons like legal, ethical, and safety aspects. Therefore, various methods have been developed and proposed to provide explanations for the decision of a single black box. The LORE approach is a promising model-agnostic method to explain the results of the black box for a particular instance understandably. But, model-agnostic methods are designed to interpret the results of a single black box model. We propose **DiRo2C** to recognize the decision differences between two binary black box classifiers, which is often necessary for practice.

Our approach uses a modified genetic algorithm from LORE to generate a synthetic balanced dataset. **DiRo2C** uses this generated diff-dataset to train a diff-classifier that recognizes the local differences close to a specific instance between the black boxes. By selecting different located instances and the generation of the diff-datasets a global explainer can be trained. It provides an explainable decision tree-based classifier where the decision tree contains the various decision rules. The decision tree is up to a certain complexity inherently interpretable. The classifier may further be interpreted by any Explainable Artificial Intelligence (XAI) approach. **DiRo2C** supports the training of a binary diff-classifier that decides if the black boxes predict different results and a multiclass predictor that predicts every possible combination of the binary black boxes results. The modified genetic neighborhood algorithm was evaluated against various other data approaches. Our simulations and experiments show that the binary classifier trained by our local modified genetic data generation approach outperforms other implemented solutions regarding accuracy and quality of detected differences.

We evaluate the performance of the classifiers, which are trained based on the various data approaches for three different datasets by applying stratified 10-fold cross-validation. In addition, we are using performance metrics like *Accuracy*, F_1*-score*, and *Pearson Correlation Coefficient*. We manipulate one black box by changing a particular feature of all instances to create differences between the black boxes. The found differences are also evaluated for correctness and whether the classifier recognizes the actual differences.

Contents

Kurzfassung **ix**

Abstract **xi**

Contents **xiii**

1 Introduction **1**
1.1 Black Box Model, Interpretability, and Explanation 1
1.2 Human Explanations 1
1.3 Problem Statement and Motivation 2
1.4 Research Questions 3
1.5 Research Method 5
1.6 Methodological Approach 5
1.7 Research Contributions 8
1.8 Structure of the Thesis 8

2 Related Work **11**
2.1 Classification Terminology 11
2.2 Interpretable Models 12
2.3 Properties of an Interpretable Model 14
2.4 Model-Agnostic Methods 15
2.5 Global Agnostic Explanators 17
2.6 Local Agnostic Explanator 18
2.7 Concept Drift Detection 23
2.8 Summary 23

3 Difference Recognition **25**
3.1 Data Approaches 27
3.2 Target Determination 35
3.3 Summary 36

4 Simplistic Genetic Neighborhood **39**
4.1 LORE's Genetic Instance Generation Approach 39
4.2 Adaptions for DiRo2C 42

4.3 Illustration of the Generated Datasets 44
4.4 Comparison of the Data Density 47
4.5 Summary 49

5 Evaluation of the Performance **51**
5.1 Experimental Setup 51
5.2 Performance Evaluation of Classifiers 61
5.3 Results and Findings 62
5.4 Summary 65

6 Modified Genetic Neighborhood and Performance Evaluation **67**
6.1 Modification of the Genetic Neighborhood Approach 67
6.2 Effects of the Modified Genetic Neighborhood Approach 70
6.3 From Local to Global Explanations 81
6.4 Performance Evaluation of the Genetic Neighborhood Approaches . . 83
6.5 Summary 85

7 Evaluation of the Detected Differences **87**
7.1 Synthetic Datasets 87
7.2 Adult Dataset 109
7.3 Summary and Findings 119

8 Conclusion and Future Work **121**
8.1 Conclusion 121
8.2 Future Work 122

List of Figures **125**

List of Tables **129**

List of Algorithms **131**

Acronyms **133**

Bibliography **135**

CHAPTER 1

Introduction

Today, Machine Learning (ML) systems make decisions in a wide variety of application areas. So-called black box models are increasingly used for this purpose, even in sensitive areas of applications or in areas where sensitive data is processed. The use of such opaque models is ubiquitous, and the results have, in many cases, crucial implications. This thesis deals with black box classifiers. For a better understanding, the introduction is intended to explain important terms, concepts, the problem statement, the research questions, the research method and methodological approach, the research contributions, and finally, the structure of the thesis.

1.1 Black Box Model, Interpretability, and Explanation

A black box predictor is a system where the internal logic isn't visible to the observer or the logic is known but not interpretable [GMR+19]. There are different definitions of interpretability regarding ML systems. Miller defines it as "the degree to which an observer can understand the cause of a decision" [Mil19, p. 14] and Finale Doshi-Velez and Been Kim as "the ability to explain or to present in understandable terms to a human" [DVK17, p. 2]. Explanations are a way for a human to obtain a better understanding [Mil19]. An explanation is, thus, an answer to a why-question [Mil19]. In the sense of scientific understanding, providing explanations leads to the fact that we gain more knowledge about the specific context [DVK17]. As Miller suggests in his work, the terms *interpretability* and *explainability* will be used interchangeably and the term *explanation* will be used for answers why the model made a certain prediction for a certain instance [Mil19]. An instance is a row, or a data point in a dataset.

1.2 Human Explanations

Miller presents in his work [Mil19] the major findings of human-friendly explanations:

Explanations are contrastive. Humans are interested in counterfactual cases. That means we want to know why a decision was made instead of another decision.

Explanations are selected. Humans are not interested in explanations that describe the complete cause of an event. It is sufficient to include one or two reasons for it.

Good explanations are general and probable. Humans are interested in explanations that contain a cause that can explain many other events. Probabilities are necessary for selecting the best causes but referring to the probability in explanations isn't effective.

Explanations are social. Explanations are part of a communication between the explainer and recipient. It is essential to pay attention to the social context and the knowledge of the participants.

1.3 Problem Statement and Motivation

As already mentioned, in many cases, the predictions of black boxes have profound consequences on the lives of individuals. Different black box classifiers are trained to provide predictions for one specific problem. Human experts need more insights into the differences to decide which classifier is a better fit for solving the underlying problem. Model-agnostic methods are suitable for getting more insights into why models predict outcomes for specific instances. Even if there is currently a dispute about whether a "right to explain" is legally binding or even exists in General Data Protection Regulation (GDPR), especially about "opening the black box" to make the decision-making process completely transparent, researching model-agnostic methods and their possibility to interpret the differences in black box models is all the more important [GF17, Mal19, SP18, WMR18]. It should be mentioned, that it is **not** always necessary to be able to interpret ML systems. Because not every system needs ad-hoc interpretability. According to the paper [DVK17] interpretability is not necessary if unacceptable decisions have no significant impact. It is also not required if the problem is well studied. That means the application is tried and tested and we have trust in the results of that decision system. Molnar argues in his book [Mol20] that e.g. an ML system for optical character recognition that detects and processes addresses from envelopes is validated and proven that such a model works correctly.
In contrast, interpretable explanations are for specific reasons necessary due to the incompleteness in the formalization of problems [DVK17]. Doshi-Velez and Kim summarized the following reasons: *scientific understanding*, *safety*, *ethics*, *mismatched objective*, and *multi-objective trade-offs* [DVK17]. Regarding ethics, discrimination is a serious problem also in the field of ML. For example, journalists investigated the Correctional Offender Management Profiling for Alternative Sanctions (COMPAS)[1] score, which is calculated

[1] Article about the investigation: https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm

to predict the criminal defendant's likelihood of becoming a recidivist [GMR+19]. They found out that the score is highly dependent on race, and as a result, people are severely disadvantaged because of it.
Due to the technical possibilities such as big data, high performance, and scalable infrastructures, we risk creating decision systems that are no longer traceable and understandable [GMR+19]. That also influences how to deal with accountability [KHB+17] and industrial liability [Kin18].
Concept (data) drifts represent another problem from a completely different point of view [Han06, Tsy04, WHC+16]: ML models are in many cases trained with historical data and optimized for a problem at a certain point in time. But the conditions and requirements for a model can change, or the measured data correspond no longer to the distribution and scaling of the initial trained model. The recognition of differences between various versions of a model supports detecting such concept drifts.
The model-agnostic method LOcal Rule-based Explanations (LORE) [GMR+18] promises confidence to provide a solution to explain the differences and ultimately make the detected differences interpretable. However, LORE and other model-agnostic methods are designed to interpret the results of a single black box model. This thesis investigates a potential solution to recognize differences between two binary black box classifiers and to enable the interpretation of the recognized differences by model-agnostic methods (see Figure 1.1).

1.4 Research Questions

This work aims to provide the Difference Recognition of 2 Classifiers (DiRo2C) method to detect differences between two binary black box classifiers. Therefore a so-called "diff-classifier" is trained to recognize the differences. One solution is to use a binary classifier that decides whether the two black box models have predicted a different or the same result. The other approach is to train a multiclass classifier to detect every possible outcome combination of two binary classifiers.

RQ 1: *"How well can the results of two binary black box classifiers be predicted using a binary diff-classifier compared to a multiclass diff-classifier to predict every possible outcome?"*

The goal is to find out if a binary black box classifier predicts the differences more accurately due to low complexity of the decision problem. It is also essential to evaluate which approach is best for training the diff-classifier. One way is to create the diff-classifier using a synthetically generic dataset. The other option is to implement it with real data of the black box models. Since it cannot be assumed that training and test data are available, the preferred solution is to prepare the diff-classifier using generic data samples. Therefore, an additional goal is to clarify whether the differences can be recognized more precisely using synthetic data for training the diff-classifier.

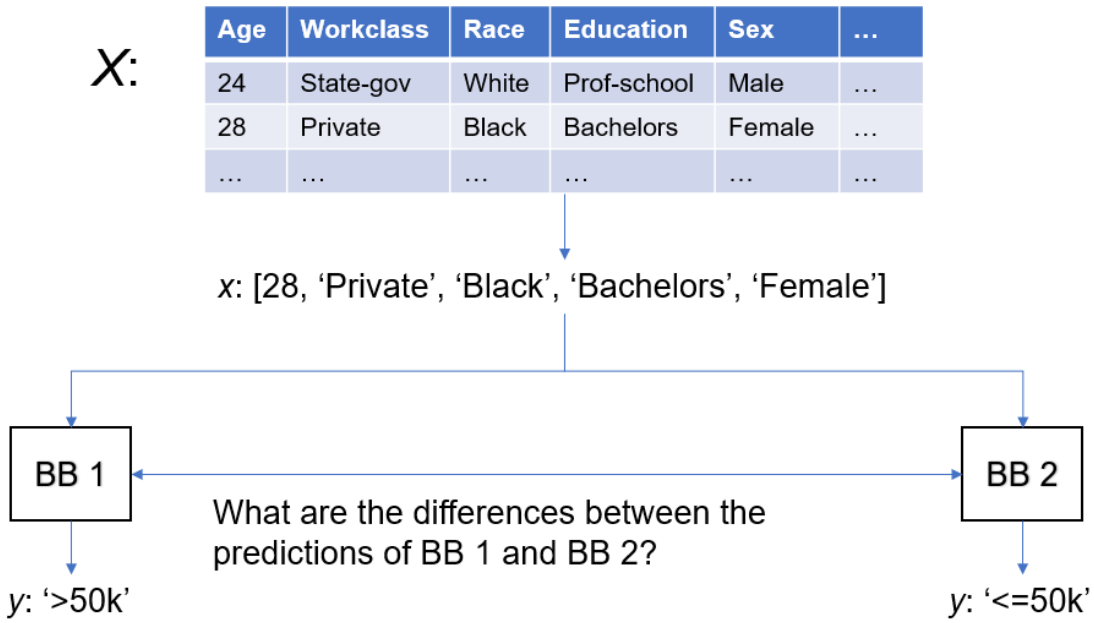


Figure 1.1: Illustration of the problem statement: In this example, two black box classifiers predict based on specific features the income class of a person, where y is the target with the possible binary classes: '<=50k' or '>50k'. The aim is to find a way to detect the specific differences between the black boxes and enabling explainability. Source of the UCI Machine Learning Repository adult dataset for this illustration: [DG17]

RQ 1.1: *"To what extent can a classifier trained on synthetic data outperform a classifier trained using a real and already existing dataset?"*

The attempt is to generate more data in the area where the predictions of the two classifiers differ. There are also different possible approaches to generate the dataset for the diff-classifier. One attempt is to generate the instances with a local approach, and the other way is to generate it globally. Local means that increased instances are generated or provided which are closer to the to be recognized instance.

RQ 1.2: *"How well performs a local approach to generate datasets for training the diff-classifier in contrast to a global one?"*

A main objective is to explore which combination of the possible approaches performs best to predict the results of the black boxes and to what extent differences are detectable. Therefore, a local synthetic approach should lead to a higher data density in the immediate area of the instance to be recognized and thus to higher coverage. The differences

found are also evaluated for correctness and whether the responsible classifier recognizes the actual (true) differences.

RQ 2: *"To what extent can differences be detected using manipulated datasets?"*

For this purpose, data instances are specifically manipulated. Subsequently, the diff-classifier is checked whether these manipulations are reflected in the decision-making basis.

1.5 Research Method

There are two basic paradigms in Information System (IS) research: behavioral and design science. The difference between those two paradigms is that in behavioral science, theories are developed and then justified. In design science, artifacts are built and then evaluated to meet the business needs [HMPR04]. Moreover, the goal of design science is to solve engineering problems (utility) and behavioral science to explain real-world phenomena (truth) [HMPR04]. An information technology artifact can be a construct, a model, a method, or an instantiation. This thesis aims to provide a method to recognize the difference between two binary black box classifiers. Therefore design science is the first choice research approach. According to Hevner, every design science research consists of three recognizable cycles: the relevance cycle, the design cycle, and the rigor cycle (see Figure 1.2) [Hev07].

1.6 Methodological Approach

This section gives an overview of the methodological approach. According to the design science research method [Hev07] the main two research activities of the methodological approach are "Development of DiRo2C" and the "Evaluation of DiRo2C". These two research activities of the design cycle are performed iteratively to assess and refine the artifact continuously. Additionally a theoretical analysis and literature review is carried out. It also explains the coding foundations to develop and evaluate DiRo2C.

1.6.1 Theoretical Analysis and Literature Review:

The analysis provides an understanding of the use of model-agnostic methods and revealing of the research gap. The review of ML algorithms (e.g., decision tree algorithm) is necessary for planning and implementing the diff-classifier strategies. LORE and especially the "Neighborhood Generation" method is a crucial concept to create local synthetic data instances. We want to use it to predict the local changes precisely. Therefore, a detailed study of that method and internal components is part of the theoretical analysis. Additionally, other local and global data generation methods are under research.

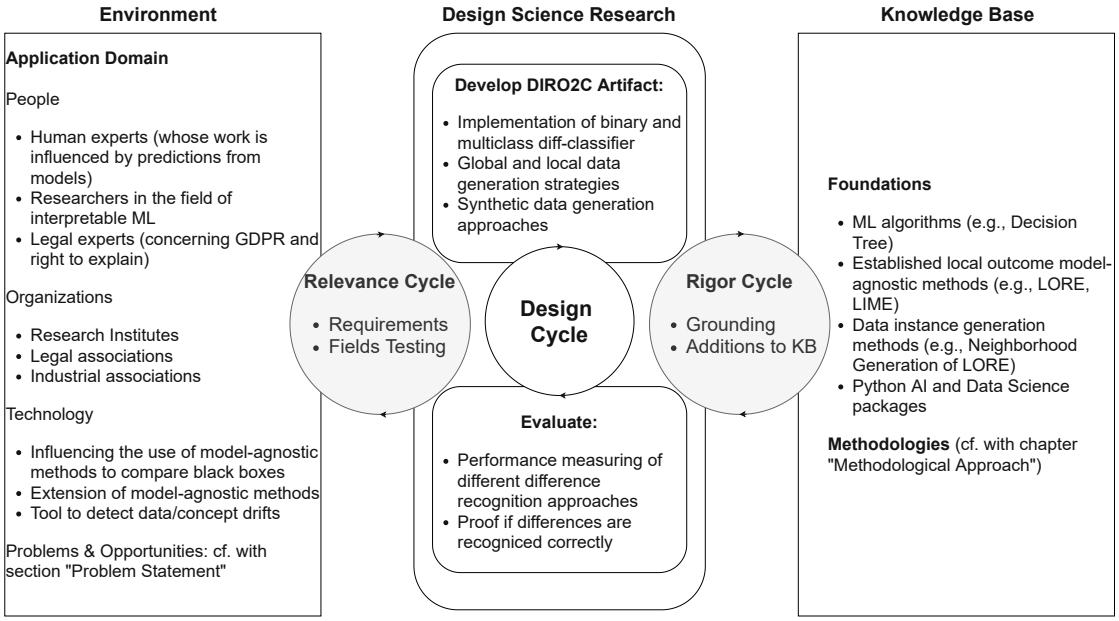


Figure 1.2: Adapted design science research cycle to summarize the research design for this thesis. Source: [HMPR04, Hev07]

1.6.2 Development of DiRo2C

The main goal of this activity is the implementation of DiRo2C and the possible approaches to detect differences. As already mentioned, one solution is to use a binary classifier, and another way is to use a multiclass classifier. In both cases, we use a decision tree for implementing the diff-classifier [BFOS84, HTF09]. In general, the training data to train the diff-classifier is produced using the available black box models. By comparing the predictions, a diff-dataset builder of DiRo2C determines the target y for the diff-dataset. We differ between a local and a global approach to generate a dataset for the diff-classifier. Local means the created data is depending on a particular single instance. It is comparable with the outcome explanation problem regarding model-agnostic methods [GMR+19]. The decisive distinction here is that the differences are computed with the predictions of the black boxes. One promising local approach to generate the training and test data focuses on the proposed *neighborhood generation* of LORE [GMR+18]. We use various implemented functions from LORE as a basis to adapt it to our problem. Therefore, the deap libary [FDRG+12] is also integrated for the adapted and modified *neighborhood generation* methods. DiRo2C uses for the binary and multiclass diff-classifier, a Classification And Regression Tree (CART) optimized algorithm[2] for implementing the diff-classifiers [HTF09, BFOS84]. We test it against the local data approach where we use the already existing black box training dataset to train

[2]Sourcecode of the algorithm: `https://github.com/scikit-learn/scikit-learn/blob/fdbaa58acbead5a254f2e6d597dc1ab3b947f4c6/sklearn/tree/tree.py#L584`

the diff-classifier. The global data generation approach is thus comparable with the model explanation problem of the "Black Box Explanation Problem" [GMR+19]. It provides one single dataset independent of a particular instance to detect the differences globally. For the synthetic global data approach, the data is generated uniformly randomly for a specific valid range of values. We also test it against another global data approach where we use the black box training dataset instead.
This phase is divided into three main steps: planning and modeling, implementing, and testing, which includes validation and verification of the implemented components.

1.6.3 Evaluation of DiRo2C

The chosen method to evaluate the implemented approaches is a lab simulation [CGH09, HMPR04, VPHB12]. To achieve reproducible results, we use controllable settings to minimize randomness. One essential objective of this phase is to find the best approach for providing the diff-classifier to predict the data instances. We measure the classifier's performance by considering the confusion matrix and evaluation metrics such as *Accuracy*, F_1-*score*, and the Pearson Correlation Coefficient (Pearson CC) [CJ20, Han12, HM15, TSK06]. In the literature, the *PearsonCC* metric is also called the Matthews Correlation Coefficient (MCC). We use the term *PearsonCC* in this work. The comparison of the metrics gives an indication of which setup performs best.

We use already existing well-known benchmark UCI Machine Learning Repository datasets [DG17] and additionally synthetically created datasets for our experiments and simulations. According to the literature, we apply a stratified k-fold cross-validator (where $k = 10$) to split the dataset and evaluate the specific metrics [Koh95, TSK06, WFH11, Won15, WY17, WY20]: This validator splits the dataset into k successive groups of samples (evenly divided), called folds. Stratified means that the instances are divided up equally that each fold preserves approximately the same percentage of each outcome class as the entire dataset. We test in 10 test runs each time with a different test instance x. Then we calculate the mean, standard deviation, min, and max for each metric.

Another essential part of the evaluation is creating test cases with synthetic and well-known benchmark datasets. The black box models are generated with manipulated datasets to evaluate the correctness. In detail, manipulated values for specific features lead to different results of the black boxes. These differences must then be recognizable in the predictions of the diff-classifier. Since we use a Decision Tree Classifier for the diff-classifier, we can analyze the created decision tree (see Figure 2.1) to evaluate the correctness. We use the machine learning extension Python library Mlxtend[3] to illustrate our results and findings [Ras18].

Therefore, we evaluate the performance of the different data generation (providing) strategies and the trained diff-classifiers. Furthermore, the detected differences of the manipulated black boxes get verified by analyzing the resulting decision trees.

[3]Python library Mlxtend: `http://rasbt.github.io/mlxtend/`

1.6.4 Code Repository and Technical Foundations

To ensure reproducible and verifiable results and ultimately scientific rigor, the code and scripts of DiRo2C and the experiment settings are publicly available in a GitLab[4] code repository. The datasets are integrated into the Python[5] scikit-learn[6] environment. Data transformations are carried out with the help of suitable packages such as NumPy[7] and pandas[8]. In addition, synthetically datasets are generated to evaluate DiRo2C. For that purpose, the scikit-learn samples generator is used[9]. We also use the Python scikit-learn open-source machine learning library for model fitting, predicting, cross-validation, and performance evaluation. The dataset is prepared to contain information regarding the differences between the black boxes. In addition, adequate metrics are selected to measure the performance of the models.

1.7 Research Contributions

Therefore, the main research contribution is the design of a method that allows the detection of the differences between two black box classifiers to provide explanations. It includes the following components: Design of two diff-classifier strategies (binary versus multiclass), local and global data generation strategies, corresponding implementations to provide datasets to train the diff-classifiers, and a synthetic data generation approach to create more data in the surrounding area where the predictions of the two black box classifiers differ. A further contribution is the implementation and evaluation of the DiRo2C artifact and if the differences are correctly recognized.

1.8 Structure of the Thesis

The thesis is structured as follows:
Chapter 2 introduces the classification terminology and gives an overview of interpretable models and related work that significantly influences our approach. Further, we explain the properties of an interpretable model. This chapter also explains different model-agnostic methods. Furthermore, we discuss why they are used and which different problems they solve. We also explain, in detail, the difference between local and global explanations and the difference between global and local explanators.
In **Chapter 3**, we focus on our approach for recognizing differences between binary black box classifiers and the main concepts of our presented approach. We explain the difference classification problem and generalize it for any number of classes for the black box base classifiers. That chapter will also introduce the dataset which we are using as a

[4]Code-Repository Environment: `https://gitlab.com/andsta/diro2c/`
[5]Python programming language: `https://www.python.org/`
[6]Python library scikit-learn: `https://scikit-learn.org/`
[7]Python package NumPy: `https://numpy.org/`
[8]Python package pandas: `https://pandas.pydata.org/`
[9]Overview of the scikit-learn sklearn.datasets module: `https://scikit-learn.org/stable/modules/classes.html#module-sklearn.datasets`

running example through our work. In addition, we explain the different data approaches to train the classifier to recognize the differences. Finally, we present the main concepts and implementations of our approach.

Chapter 4 explains in detail the "Genetic Neighborhood" approach of LORE and how we adapt this approach for our local genetic neighborhood approach to recognize differences between two binary black box classifiers. We explain the used fitness functions of LORE that are an integral part of genetic algorithms. For comparison, we show the datasets of the different data approaches. We also present the data density plots of each approach to better explain the differences between the global and local approaches.

Chapter 5 describes how we measure the performance of the trained classifiers by the various data approaches. We also present the experimental setup. In detail, the used benchmark datasets, the preprocessing of the datasets, and the black box training and manipulation to force differences between the black boxes. We also explain which performance metrics are used for evaluating the performance. Finally, we present the results and discuss our findings. We also explain why the local genetic neighborhood approach of DiRo2C may be too simplistic.

In **Chapter 6**, we focus on presenting our modification of the initial genetic neighborhood approach. We explain the modified fitness function and implementation of the algorithm. After that, we show the effects of the modified local genetic neighborhood approach by using our running example and additionally a Gaussian quantiles dataset. Again, we also use data density plots to illustrate the differences between the genetic neighborhood data generation approaches. After all, we discuss how we want to solve the ultimate goal to explain the global differences between two binary black box classifiers using the local modified genetic neighborhood approach to generate various diff-datasets. Finally, we present the measured performance results of the modified approach compared to the simplistic genetic neighborhood approach and discuss our further findings.

Chapter 7 presents our evaluations if the detected differences are recognized correctly. Therefore, we use simple two-dimensional synthetic datasets to observe if and how forced manipulations are recognized. In addition, we want to show how our approach can handle non-orthogonal decision boundaries. We also use a well-known benchmark dataset to evaluate the recognized differences for correctness. We present data manipulation scenarios with various defined hypotheses and how the forced manipulations should affect the trained classifier to recognize the differences.

Chapter 8 summarizes the main parts, findings, and contribution of this thesis. Furthermore, we summarize the limitations of our work and discuss potential further works.

CHAPTER 2

Related Work

This chapter introduces the terminology and core concepts in terms of interpretability. It gives an overview of related topics and work that significantly influences our approach. According to the literature, we differ between approaches that explain the global and local behavior of a black box model. We use local data approaches to predict the local differences between the black boxes. Besides, we implement data approaches for DiRo2C to explain the global behavior of the black box models and thus to detect not only the local differences between them. This chapter also focuses on model-agnostic methods that provide a function f (especially an interpretable model) to explain any black box. In addition, rule-based approaches and, finally, the concept drift detection are discussed.

2.1 Classification Terminology

According to the work of Guidotti et al., we define the classification problem as follows [GMR+18, GMR+19]: a classifier (predictor) is a function (see Equation 2.1):

$$c : \mathcal{X}^{(m)} \rightarrow \mathcal{Y} \tag{2.1}$$

that assigns an instance x to a classification (also called decision, prediction, outcome or target) y in a target space $\mathcal{Y}$. Where x depends on a feature space $\mathcal{X}^{(m)}$ with m features. A classification y is predicted by c (see Equation 2.2):

$$c(x) = y \tag{2.2}$$

and $c(X) = Y$ denotes $\{c(x) \mid x \in X\} = Y$, where X is the set of all inputs. An instance is defined as a set of m attribute-value pairs: (a_i, v_i), where a denotes the feature (attribute) and v is a continuous or categorical value. Besides, a predictor can be any

decision system. In the following, c represents an interpretable classifier (cf. Section 2.2) and b denotes a black box predictor. Therefore, we will write (see Equation 2.3):

$$b(x) = y \tag{2.3}$$

Occasionally we differentiate in our work between the actually assigned outcome class to train the classifier and the predicted target of the classifier. In that case, y denotes the actually assigned target and $\hat{y}$ the predicted target. According to the summary of Guidotti et al. [GMR+19] the following methods or algorithms are designated black boxes: Neural Network (NN), Tree Ensemble (e.g. Random Forest), Support Vector Machine (SVM), Deep Neural Network.

2.2 Interpretable Models

Certain types of ML models have the property that the resulting decisions are inherently interpretable. Interpretable models are also called white boxes. According to the state of the art, *decision trees*, *rules*, and *linear models* are such interpretable models [Fre14, GMR+19, HDM+11, Mol20, RSG16b].

2.2.1 Decision Tree

A decision tree-based system uses a structured graph that is composed of internal nodes [BFOS84, HTF09, Qui86]: Decision tree model algorithms calculate, several times, threshold values for the features to divide the data according to the cutoff criteria. A feature is an attribute or a column of a dataset. Thus, the features are the inputs to classify an instance. An internal node represents a condition for a feature, and the corresponding value is tested against it. The leaf nodes represent the class label and decide the result of the prediction (see Figure 2.1). One path from the root to the leaf node can be considered as a classification rule and has a defined outcome.

2.2.2 Decision Rule

Therefore, another suitable approach to generate interpretable systems is to use a decision rule (see Equation 2.4) which consists of one or more conditions and the corresponding outcome [FGL12, FW98, Mol20]:

$$IF\ condition_1\ \wedge\ condition_2\ \wedge\ ...\ \wedge\ condition_n\ THEN\ outcome \tag{2.4}$$

A decision rule can be interpreted as one path from the root to a leaf node of a decision tree. Thus, a set of rules can be derived from a decision tree [FW98, Qui87a, Qui87b]. The logical conjunction of conditions for specific features in the if-clause is comparable to a certain path with the corresponding conditions (threshold values) and the outcome is comparable to the leaf node which represents the class label. In the field of rule-based

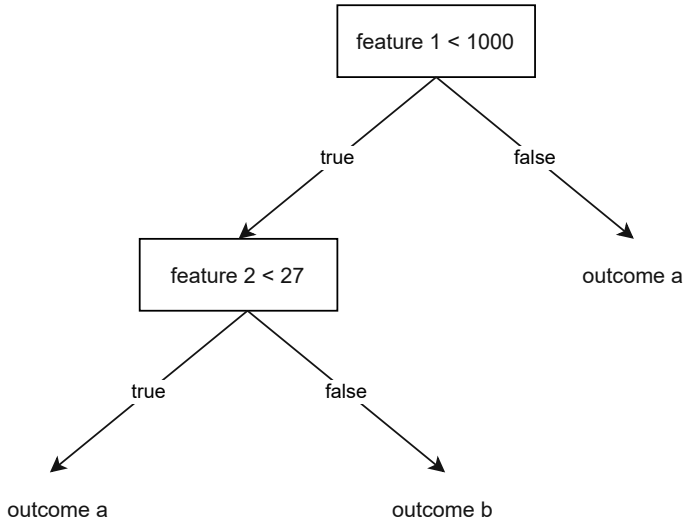


Figure 2.1: Example of a binary decision tree

IF Respiratory-Illness = Yes AND Smoker = Yes AND Age >= 50 THEN Lung Cancer

IF Risk-LungCancer = Yes AND Blood-Pressure >= 0.3 THEN Lung Cancer

IF Risk-Depression = Yes AND Past-Depression = Yes THEN Depression

IF BMI >= 0.3 AND Insurance = None AND Blood-Pressure >= 0.2 THEN Depression

IF Smoker = Yes AND BMI >= 0.2 AND Age >= 60 THEN Diabetes

IF Risk-Diabetes = Yes AND BMI >= 0.4 AND Pro-Infection >= 0.2 THEN Diabetes

IF Doctor-Visits >= 0.4 AND Childhood-Obesity = Yes THEN Diabetes

Figure 2.2: Comprehensible interpretable decision set, which enables applying rules independently. Source: Lakkaraju et al. [LBL16]

approaches, some interesting concepts have been presented to be able to interpret and explain the decisions of models. Letham and Rudin proposed in their paper a method to build predictive models in the form of sparse decision lists that contain "if-then" statements [LRMM15]. Lakkaraju, Bach, and Leskovec present an approach to describe the decision boundaries between classes using decision sets (see Figure 2.2) [LBL16]. In conclusion the following papers propose rule-based approaches: [LBL16, LRMM15, MVED17, WR15].

Decision trees compared to rule-based systems have decisive advantages and disadvantages and are discussed controversially in the literature [FGL12, Fre14, Qui87a, Riv87]: Decision trees can represent the nodes and the included conditions graphically. The hierarchical structure helps to identify the relative importance of an attribute. The closer the node is to the root, the more important is the regarding feature for the prediction. In contrast, decision rules, which represent the basis for decision-making textually, are more compact and expressive.

2.2.3 Linear Model

Linear model approaches also offer the possibility to interpret the decisions of the trained ML systems [HMG+14, Mol20, RSG16b]. Equation 2.5 shows how the model predicts the outcome y for a particular instance x of X [HTF09]:

$$y = \beta_0 + \sum_{i=0}^{m} \beta_i x_i \tag{2.5}$$

Where x_i represents the input for the i-th feature of the instance x and β the regarding weights or coefficients. With the help of the weights and analysis of the feature importance (see Figure 2.3), the decisions of a linear model are interpretable [GMR+19, Mol20]. The sign and magnitude for the coefficient reveal the influence the feature has on the decision [GMR+19, HMG+14].

2.2.4 Interpretability

In conclusion, the more complex the model is (e.g., a too large set of rules or deep decision tree), the more likely it becomes that it is not humanly understandable any longer [Fre14, GMR+19, HDM+11]. Lipton stated in his work, that a model is only interpretable if a human can capture the entire model at once [Lip17]. He even argues that an interpretable model trained with high dimensional data is in certain circumstances less transparent than a compact NN.

2.3 Properties of an Interpretable Model

Interpretable models must have the property that the decisions are comprehensible and explainable. Guidotti et al. summarize in their work the most important and desired criteria for interpretable models: *Interpretability*, *accuracy* and *fidelity* [GMR+19](cf. also with [DVK17]).

Interpretability: The decisions of interpretable models must be human-understandable. As already mentioned in Section 2.2, the complexity of the model is crucial so that the basis of the decision is understandable [Fre14, HDM+11]. The term *interpretability* is to be understood in the same way as the term *comprehensibility*.

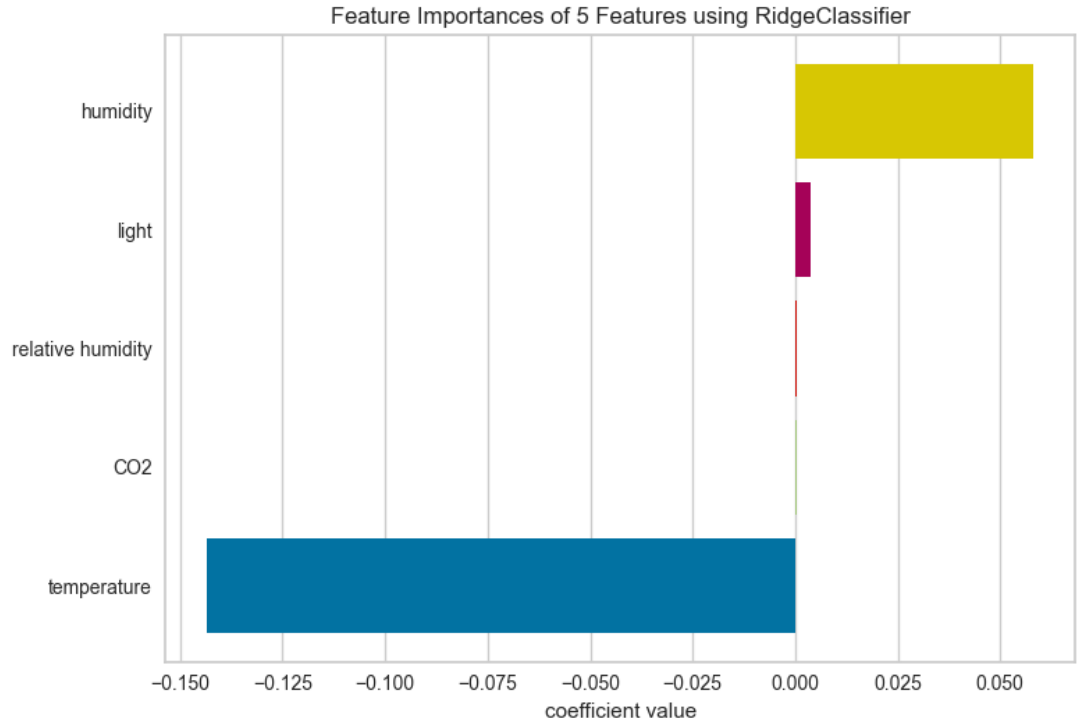


Figure 2.3: Feature importance of a linear model. Example of a binary classification for room occupancy, where the targets are 1 for "occupied" and 0 for "not occupied". Source of dataset: [CF16]

Accuracy: Interpretable models have to predict unseen data or instances accurately. High accuracy is the goal if the explanations of the interpretable model are considered for predictions and not the decisions of the black box model. The model's performance can be measured with suitable evaluation metrics such as *Accuracy* and F_1-*score* [Han12, HM15, TSK06].

Fidelity: The explanations of the interpretable models must accurately approximate the prediction of the black box. Fidelity gives information about how well the model imitates the behavior of the entire black box. The measurement is carried out again via *Accuracy* and F_1-*score*, but in regards to the result of the model.

2.4 Model-Agnostic Methods

Model-agnostic methods are suitable approaches to interpret predictions of black box models and provide solutions for the "Open the Black Box Problem" [GMR+19]. Different methods help to separate the explanations from the ML model [RSG16a]. That means an agnostic explanator is not only compatible for one specific type of ML model like NN or

a SVM [GMR+19]. Desirable characteristics of model-agnosticism are: *model flexibility*, *explanation flexibility*, and *representation flexibility* [RSG16a].

Model Flexibility: Interpretable models provide not for all problem-solving tasks sufficient understandable explanations. In the case of a complex task (e.g., predicting the sentiment of a sentence [RSG16a]), the model and, finally, its decisions would be no longer understandable by humans [Fre14]. Thus, model flexibility is fulfilled if the interpretation is independent of the underlying machine learning model. The separation of interpretability allows using any machine learning model to provide interpretable explanations.

Explanation Flexibility: Regarding the application and the problem-solving task, different forms of representation are needed for explaining. For some problem-solving tasks, a graphical illustration of the feature importance is helpful, in other cases a linear formula [Mol20]. Ribeiro et al. describe *explanation flexibility* using the following example: By explaining why an image is classified into a particular category, it could be sufficient to know which part of the image is most responsible for the classification [RSG16a]. Therefore, *explanation flexibility* ensures that a model-agnostic method is no longer bound to a specific form of explanation. In addition, users with different specialist knowledge require alternate explanations. *Explanation flexibility* also concerns the complexity or granularity of the explanation. For some cases and certain user groups, complex or multiple decision rules are desirable, and in some cases, simple or few rules are more suitable. Ribeiro et al. explain that most interpretable models are restricted in providing flexible explanations [RSG16a]. E.g., the approach of Kim et al. by using prototypes for the explanations [KRS14], of Letham et al. by providing a set of rules [LRMM15], or of Caruana et al., which use line graphs [CLG+15].

Representation flexibility: Ribeiro et al. argue in their work that in areas where ML problems relating to video, image, and text have to be solved, many features are used, which are not per se human-understandable without changing the representation of the feature [RSG16a]. E.g., unsupervised learning is used to provide word embeddings (cf. [MSC+13]). Ribeiro et al. further stated that even if an interpretable model is used to return explanations for word embeddings, the feature is still non-interpretable. Therefore, *representation flexibility* implies that the model-agnostic approach can represent a feature differently as the model has been trained. That means model-agnostic methods that fulfill *representation flexibility* can provide explanations by replacing the non-interpretable features with interpretable ones.

The **"Black Box Explanation Problem"** is divided into *model explanation*, *outcome explanation*, and *model inspection* [GMR+19]:
Model explanation approaches deliver global explanations to interpret the complete decision-making basis of the classifier (see Figure 2.4). Thus, the provided interpretations reflect the inner logic of the black box.

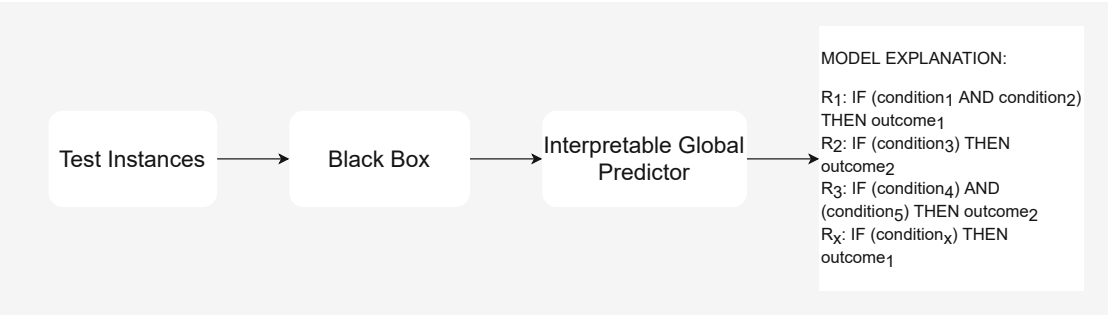


Figure 2.4: Model explanation problem: Providing an interpretable global predictor for example in form of decision rules. Source: Guidotti et al. [GMR+19]

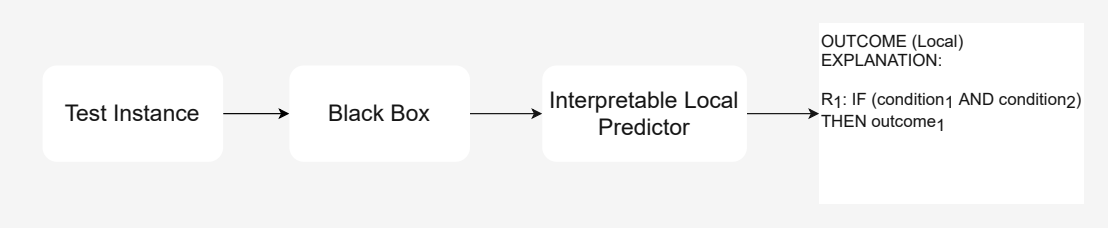


Figure 2.5: Outcome explanation problem: Providing an interpretable local predictor to explain the prediction for a particular instance. Delivers only those rules which are responsible for classifying that instance. Source: Guidotti et al. [GMR+19]

The challenge regarding the **outcome explanation** problem is to provide a local explanation of why the black box predicted the result for a particular instance (see Figure 2.5). The **model inspection** problem consists of delivering textual or visual representations to get an overview of some specific property and why the black box predicts certain results for particular value ranges.

According to Guidotti et al. and their work [GMR+19], one method to solve the **model inspection problem** is Individual Conditional Expectation (ICE) [GKBP15]. It is equivalent to the Partial Dependency Plot (PDP) method with the significant difference that ICE explains the effect of changes of single instances.
It should be mentioned that any interpretable model (cf. Section 2.2) can be used for an agnostic explanator [GMR+19, Mol20].

2.5 Global Agnostic Explanators

Global agnostic explanators provide a solution to the **model explanation problem** [GMR+19]. There are different approaches to interpret black boxes globally independent of the underlying black box model. The method of Lou et al. [LCG12] was one of the first to provide an agnostic solution. It returns the feature importance and additionally the shape function for the explanation. A shape function illustrates the

characteristics of a function by using a plot. Using the plot, the linear and the non-linear properties together with its shape linearities can be analyzed. They use and extend Generalized Additive Models (GAMs) (cf. [Has90]) to interpret splines, regression trees, or tree ensembles. Guidotti et al. claim in their work [GMR+19] that GAMs are seen as the first choice (compared to other approaches) for interpretable explanations when only univariate terms are considered. Another method is called GoldenEye [HPB+14]. GoldenEye implements an iterative algorithm (based on data randomization) to find features that contribute significantly to the prediction. The Partition Aware Local Model (PALM) [KW17] method's objective is to debug the machine learning of the black box by capturing the structure of the dataset. PALM approximates the behavior of the black box by using a decision tree as a meta-model. The meta-model is responsible for partitioning the training data. PALM also uses a set of sub-models to mimic the behavior for each partition. Thus, a user can interpret the detected rules by tracking the path and assign problematic test examples to the responsible train data. Guidotti et al. claim in their work [GMR+19] that PALM has also the property of an explanator agnostic method. The PDP [Fri01] and the Accumulated Local Effects (ALE) [AZ19] approaches are used to provide global explanations to be able to show how individual features affect the prediction of the black box model on average. The main difference between these two methods is that PDP describes the marginal effect on the predictions and ALE the average of the changes in the predicted outcomes. Therefore ALE has an important advantage when the features are correlated.

2.6 Local Agnostic Explanator

Local agnostic explanators provide a solution to the **outcome explanation problem** [GMR+19]. There are various local agnostic explanatory approaches. We explain in this section especially the LORE and Local interpretable model-agnostic explanations (LIME) method. Similar to those approaches, DiRo2C also uses a surrogate model to predict the decisions of the black boxes. A surrogate model tries to approximate the predictions of an existing ML model. These two approaches imitate and further explain the local behavior for a particular instance.

The **LIME** method approximates the results of the black box by using local surrogate models [RSG16b]. Further, LIME provides explanations by using interpretable linear models (cf. Section 2.2). The calculated feature importance of the linear models represents the explanation (see Figure 2.3 and 2.6). The method generates random data in the neighborhood of the to be explained instance to train the linear models. LIME explains the local behavior of the black box for a particular instance x. Ribeiro et al. further provide sparse linear models to reduce the weights and thus the features used in the explanations. When using the so-called "SP-LIME" method, the user has to choose the length of the explanation. The authors stated that LIME uses a measure $\mathcal{L}$ to determine how faithful the linear model approximates the local behavior of the black box model. The black box model is explained by LIME locally, which is defined by a

```
Instance:
   income_class  age workclass education marital-status  ...     sex capital-gain capital-loss hours-per-week  native-country
20        <=50K   46   Private      11th      Separated  ...  Female           0            0             40   United-States

[1 rows x 13 columns]

Feature importance:
capital-gain <= 0.00 -0.501
relationship=Not-in-family -0.083
hours-per-week <= 40.00 -0.083
education=11th -0.078
occupation=Machine-op-inspct -0.051
37.00 < age <= 48.00 0.05
sex=Female -0.048
workclass=Private -0.044
race=White 0.036
marital-status=Separated 0.008
```

Figure 2.6: LIME explains in this example the local behavior of an instance by using the calculated feature importance of the approximated linear models. “income_class” is the target y. Source of the adult dataset: [DG17]

proximity measure. The term “faithful” corresponds to the term “fidelity” in the context of [RSG16b]. The authors describe that LIME measures the distance which is a crucial factor of $\mathcal{L}$ between the to be explained instance x and close to x generated instances. Those synthetic instances are generated close to x uniformly at random. The distance is measured by applying an exponential kernel defined on a distance function (e.g., cosine distance for text, $L2$ distance for image) to determine the locality proximity measure. Ribeiro et al. claim that to ensure interpretability and local fidelity, LIME minimizes the $\mathcal{L}$ measure plus a complexity measure for different generated explanations. In the case of linear models, the complexity measure is controlled over the number of non-zero weights. Figure 2.7 shows the functionality of LIME using a toy example [RSG16b]. The function of the black box classifier is represented by the blue and pink background (two different outcome classes). The to be explained instance is marked with a red, bold cross. The figure also shows the randomly generated instances close to the instance to be explained. LIME predicts the outcome for the generated instances by applying the black box classifier and weighs them by using the locality proximity measure. Ribeiro et al. explain that the dashed line represents the minimized explanation that is locally faithful. Ribeiro et al. claim that the complex function of the black box classifier cannot be approximated global adequately by a linear model. Therefore, they also mentioned that it is not guaranteed that the explanation is globally faithful as presented in the figure.

Like LIME, the anchors’ approach [RSG18] uses a perturbation-based algorithm to create randomly so-called anchors with the highest coverage. Instead of using a linear model, the authors use a decision rule in the form of a “IF-THEN” statement for the explanation. Anchors are reusable for locally similar instances.

Compared to LIME, there are also other approaches that compute the feature importance for a particular instance x to provide local explanations. Strumbelj et al. propose an

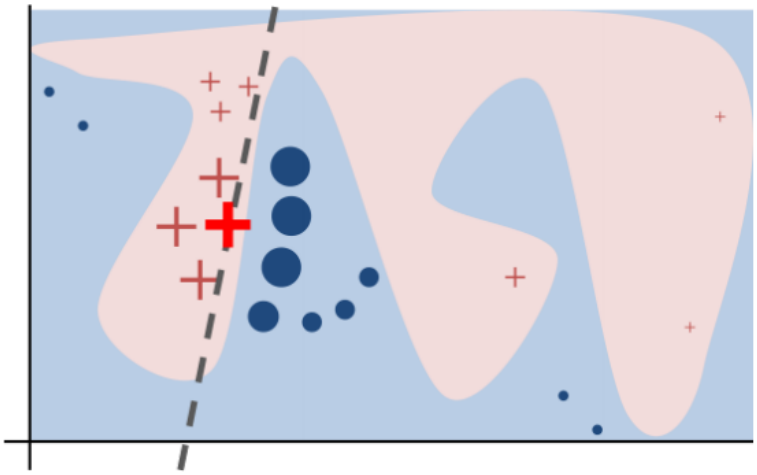


Figure 2.7: Toy example to present intuition for LIME. Source: [RSG16b]

approach that explains the prediction by highlighting the contribution of a particular feature value on the decision of the black box [ŠKŠ09, ŠK10]. The Interactions-based Method for Explanation (IME) method provides the feature importance of the black box's decision for a particular to be explained instance. Further, Strumbelj et al. claim that they use only the input and output of the black box to calculate the shift of the black box's decisions and transform it into contributions (cf. feature importance) of individual feature value on the decision [ŠK10]. The authors describe that they apply for computing these contributions known concepts from the coalitional game theory [ŠK10]. A similar approach to LIME is the Model Explanation System (MES) [Tur16] of Turner et al. providing explanations using a Monte Carlo algorithm. They implement a "pre-computation" phase and a scoring system to find the best explanation. Turner explains that for each explanation the method is finding by using a "precomputation" phase the optimal threshold and its corresponding score [Tur16]. Then all optimized scores for each explanation are compared to find the highest score. The explanation contains a logical statement that expresses the reason for the decision of the black box. SHapley Additive exPlanations (SHAP) [LL17] is in principles proposed to explain individual predictions. SHAP values describe the influence of each feature on the prediction for a particular instance. But SHAP, which is derived from the coalitional game theory method Shapley Values [Sha53], provides also extensions for global interpretation methods.

In contrast, **LORE** uses logical rules and a set of counterfactuals to provide local explanations to explain the prediction for particular instances [GMR+18]. Counterfactuals and logical rules are human-friendly explanation concepts. Miller summarized the characteristics of human-interpretable explanations in his work [Mil19] (cf. Section 1.2). A counterfactual describes how the input of certain features must be modified to change the outcome [WMR18, Mol20]. To create counterfactuals many different approaches

and implementations are already established to provide explanations. Wachter et al. propose one method to provide counterfactuals in their work [WMR18]. The extended implementation of Van Looveren and Janis Klaise uses class prototypes for providing better interpretability [LK20]. LORE uses a decision tree to approximate the results of a black box and is user-parameter-free [GMR+18]. User-parameter-free means that no further inputs are required by the user to create an explanation for a particular instance. LORE uses a genetic algorithm approach (we explain the LORE method in detail in Chapter 4). The method generates the neighborhood close to an instance x to be explained by applying a genetic evolutionary approach, which is described in the work of Bäck et al. [BFM00]. At first, we want to show how differently the method generates the neighborhood compared to a random data generation approach. Figure 2.8 illustrates two different neighborhood data generation approaches. The figure is originally shown in the following work: [GMR+18]. Guidotti et al. explain that the figure shows an example of a neighborhood generation based on a black box random forest model with a bi-dimensional feature space. The right figures show the genetic neighborhood data generation of LORE. The left figures show a uniformly random data generation for comparison. The bottom figures show additionally the data density around the starred instance. The starred datapoint highlights the instance x to be explained. The bottom figures show the data density using a color scale (by a yellow gradient). The genetic neighborhood approach generates, in contrast, to the uniformly random data generation approach, increased data around the instance x to be explained. The authors claim that the data density close to x is a key factor for training local interpretable predictors and helps create an accurate local explanation of why the black box predicts the specific outcome. After clarification with Guidotti et al. (authors of [GMR+18]), we also can explain why the figure shows darker green and purple stripes in some areas. Those darker stripes indicate areas in which the classifier is more confident of a certain outcome. We assume that the classifier in the white areas is less confident (compared to other areas) of a certain outcome.

LORE creates an explanation e (see Equation 2.6) by extracting the used decision tree as local explainer in the following form [GMR+18]:

$$e = \langle r = p \rightarrow y, \Phi \rangle \tag{2.6}$$

Where r represents a decision rule that describes the reason for predicting: $y = c(x)$. Φ represents counterfactuals that give information about which changes for the respective features lead to an opposite prediction. Guidotti et al. provide the following example to create for a loan request for an customer x (see Equation 2.7) an explanation e (see Equation 2.8) [GMR+18], whereby x represents a particular to be explained instance:

$$x = \{(age{=}22), (job{=}none), (amount{=}10k), (car{=}no)\} \tag{2.7}$$

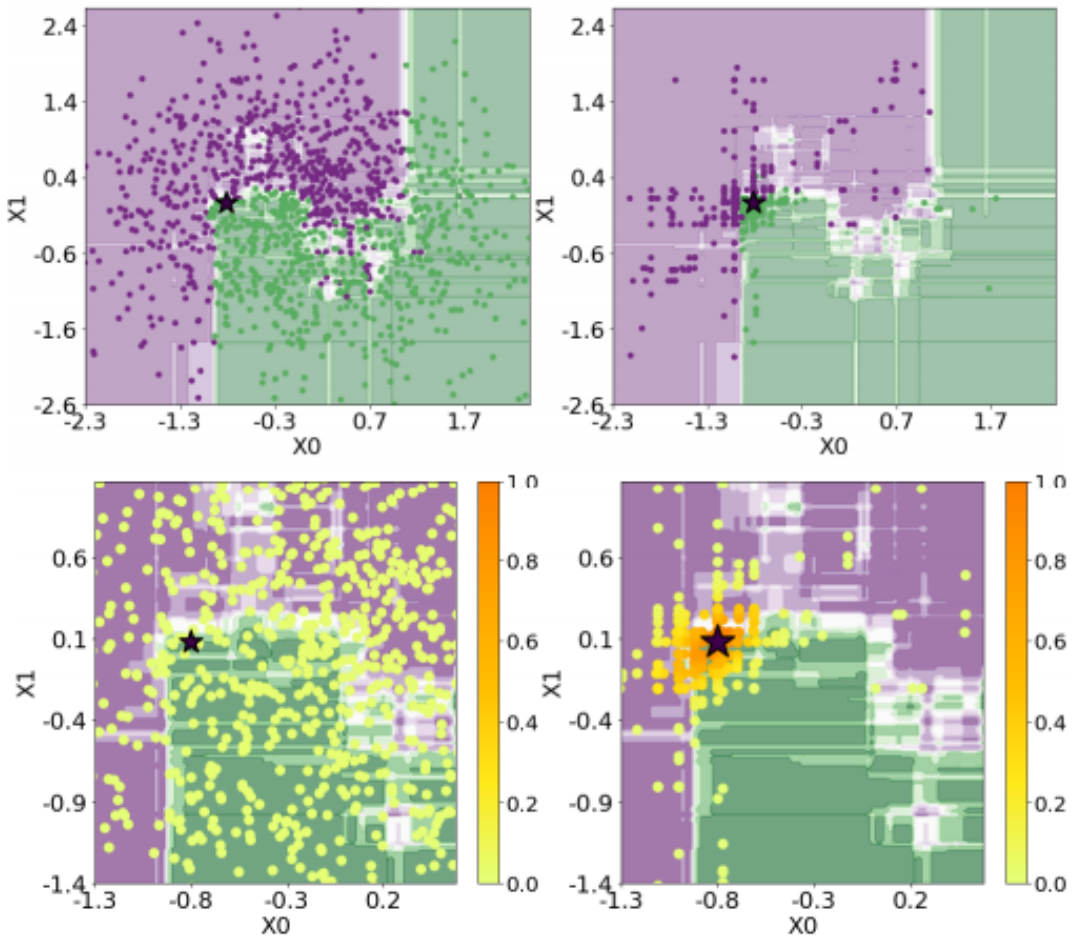


Figure 2.8: Genetic neighborhood data generation of LORE vs. uniformly random data generation. Decision boundary of a random forest black box: purple and green outcome. The starred datapoint highlights the to be explained instance. The left figures show a uniformly random data generation, and the right figures the genetic neighborhood data generation of LORE. The bottom Figures show for comparison the data density around the starred instance. Source: [GMR+18]

$$\begin{aligned} e = \langle r = \{age \leq 25, job = none, amount > 5k\} \rightarrow deny, \\ \Phi = \{(\{age > 25, amount \leq 5k\} \rightarrow grant), \\ (\{job = clerk, car = yes\} \rightarrow grant)\rangle \end{aligned} \tag{2.8}$$

Guidotti et al. describe in their paper [GMR+18] the limitations of LIME [RSG16b] and the anchor's approach [RSG18]: They argue that a random generation of the neighborhood is not focused on generating increased instances nearby the local environment of the

instance to be explaining. Thus, this approach is not advantageous to achieve a higher data density closest possible to the particular instance (see Figure 2.8). Furthermore they stated, it is not beneficial that a user has to specify: the number of features in [RSG16b] and the level of precision in [RSG18]. Therefore, we have **chosen the genetic neighborhood generation approach of LORE** as the basis for our data generation approach to recognize the local differences between the black boxes. Furthermore, we apply the "Genetic Neighborhood" approach of LORE to generate a dataset dependent on a particular instance to train the diff-classifier. In our approach, we use a decision tree classifier to predict the differences between black boxes. We modify the genetic neighborhood approach of the LORE method to generate new instances depending on the outcome of the black boxes.

2.7 Concept Drift Detection

In the field of concept drift research, some interesting papers present an approach to detect if drifts are occurring. Most of the literature focuses on detecting a drift rather than explaining the differences (cf. [DB18, GMCR04, LZL14, WHC+16]. Apart from the work of Demar and Bosni that proposes a drift detector to recognize concept drift using model explanation [DB18]. The recognizer is compatible with an arbitrary classification algorithm. Demar et al. use the IME approach [ŠKŠ09, ŠK10] to recognize redundancies and interactions of features. Demar et al. claim that IME has the needed characteristics of being resistant to noise and enables interpretable macro (cf. *model explanation problem*) and micro (cf. *outcome explanation problem*) visualization of concept drifts. This method highlights the influence of a feature on a particular prediction of the ML model to interpret the global behavior (cf. Figure 2.3). They argue further that it shows the contribution of specific feature values for interpretable micro-visualization.
In contrast, our approach provides a diff-classifier that is interpretable by any model-agnostic method.

2.8 Summary

In this chapter, we have defined the classification terminology, which is used in this work. We have proposed, according to the literature, accepted interpretable models. We have discussed concepts in terms of interpretability and different approaches to explain the decision of black boxes. We also have discussed the differences between local and global explanations and present approaches as examples. Further, we have presented global and local agnostic explanators, related topics, and in detail LORE and LIME that significantly influence our approach. We have explained why we have chosen the genetic neighborhood data generation of LORE for our purpose and designed DiRo2C based on it. The next chapter will describe the approach of DiRo2C to detect differences between the black boxes. We will explain the difference classification problem between two black boxes and also propose data generation (providing) approaches to train the diff-classifier. Further, we will explain our approach to train a binary classifier to detect

whether the two binary black boxes A and B predict the equal outcome or not and a multiclass classifier that can predict every possible outcome of the two binary black boxes A and B. We also will generalize the problem to any number of classes.

CHAPTER 3

Difference Recognition

This chapter focuses on the approach for DiRo2C to detect differences between the binary black boxes and the main components. The differences between the black boxes are recognized by training a decision tree-based classifier. According to RQ 1, the aim is to investigate how a binary classifier performs in contrast to a multiclass classifier. Therefore, DiRo2C provides a binary and multiclass classifier for recognizing differences. To illustrate the *difference classification problem* between two black boxes, we provide Figure 3.1. The left plot and the middle plot show the two datasets of the black boxes A and B and the classified instances. Each instance of black box A and B (shown in the plot as a datapoint) is assigned to a particular class. The positive "1" class of black box A and B is marked with a red "plus" sign. The negative "0" class of black box A and B is marked with a blue "minus" sign. The yellow line marks the boundary of class membership for an instance. The plots show two features x_1 and x_2. The dataset for black box A is vertically divided in half. For the dataset of black box A, the x_1 feature decides whether the instance is classified as a positive or negative class. The dataset of black box B has a more complex class distribution. The plot shows on the x_1 axis three manipulated shifted (staggered) boundary sections. Starting from the bottom up, the first boundary section is the same as for black box A, the second is shifted to the right in favor of the positive class and the third is shifted to the left in favor of the negative class. The plot on the right shows the differences between the datasets of black box A and B. Further, it shows the resulting possible classes that represent the different combinations of the outcome classes of black box A and B. The different classes are marked with "00", "11", "10", and "01". The class boundaries are again marked with a yellow line. Class "00" marks the area where the instances of black box A and B are both assigned to the negative class. Class "11" marks the area where the instances of black box A and B are both assigned to the positive class. Class "10" marks the area where the instances of black box A are assigned to the positive class, and the instances of black box B are assigned to the negative class. Class "01" marks the area where the

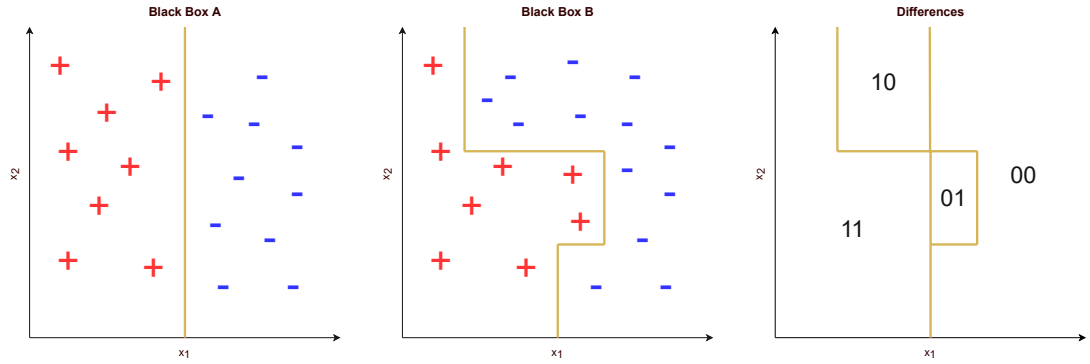


Figure 3.1: Difference recognition illustration of two datasets for black box A and B, used as a template for our running example. The dataset of black box B shows a more complex boundary between the negative and positive class. The right figure shows the differences between black box A and B and the different possible labels.

instances of black box A are assigned to the negative class and the instances of black box B are assigned to the positive class. In the case of the binary diff-classifier, the classes "11" and "00" are combined and classified as "no diff" or y=0, and the classes "10" and "01" are combined and classified as "diff" or y=1. We will use similar two-dimensional synthetic classification datasets for our running example throughout the work. Figure 3.2 shows another example the so-called "diagonal example" in the following. The dataset of black box A remains unchanged compared to the example shown in Figure 3.1. But, the dataset of black box B is now diagonally divided into two classes. The classification of the instances is dependent on both features x_1 and x_2. Again, the plot on the right shows the differences between black box A and B and the resulting possible *difference-classes* "11", "00", "10", and "01". The instances are classified differently in the lower-left corner and the upper right corner. In the case of the multiclass diff-classifier, each combination of black box A and B can be classified. We use this second example above all to show how our decision tree-based diff-classifier can handle non-orthogonal boundaries. We explain the diagonal example dataset in detail in Chapter 7.

The *difference classification problem* to recognize possible differences between two black box classifiers that classify an instance into cn possible classes can be generalized, as shown in Figure 3.3. The figure shows the difference classification into cn^2 difference-classes. The various difference-classes can be presented by a $cn \times cn$ matrix depending on the corresponding classes of black box A and B. The class labels of black box A are labeled with "cl_a", and the class labels of black box B are labeled with "cl_b" in the figure. The cell values of the matrix denote the various *difference-classes*. The yellow marked diagonal indicates classes where the two black boxes predict the equal outcome class. For DiRo2C, we focus exclusively on the 2 (binary) class problem and the 4 (multiclass) class problem. The 2 (binary) class differences classifier divides the classes whether black box A and black box B have predicted the same or a different result. The 4 (multiclass) class differences classifier divides the classes to predict all possible

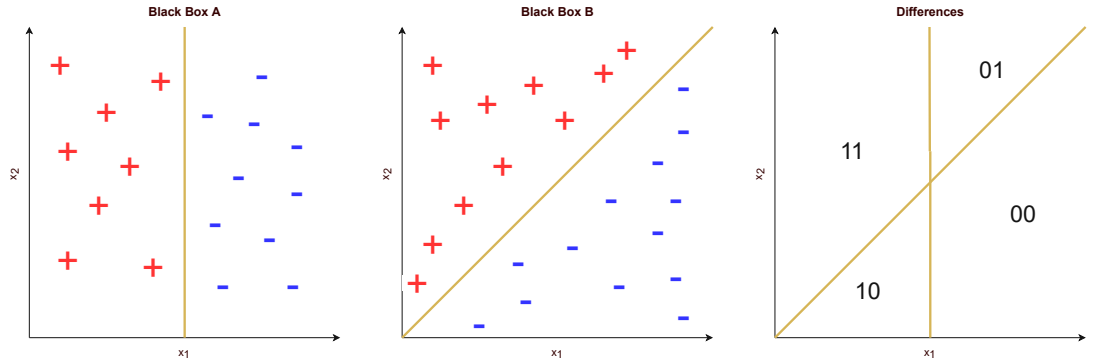


Figure 3.2: Another difference recognition example of two dataset for black box A and B. The dataset of black box B shows a diagonal boundary between the positive and negative class. The right figure shows the differences between black box A and B and the different possible labels.

combinations of the binary black box classifiers.

DiRo2C implements for both cases a CART optimized algorithm to predict the differences. As already mentioned, a multiclass classifier has the decisive advantage that it can predict the various possibilities of differences. The hypothesis, however, is that it performs worse due to its higher complexity. This chapter explains the different components and processes for both classifier approaches. So that a diff-classifier can be trained at all, the diff-dataset Z_{Diff} must first be generated or provided, and then the outcome y must be determined (see Figure 3.4). According to RQ 1.1, we want to test and evaluate if a classifier trained on synthetic data outperforms a classifier trained using the original training dataset. That means, in this case, we do not generate a dataset but provide an already existing dataset. The existing dataset is provided by merging the initially used datasets to train black box A and B. According to RQ 1.2, we want to find out to what extent a local approach, to recognize differences, can outperform a global one to generate the diff-dataset. Therefore, we explain the different data approaches.

DiRo2C can detect differences between black boxes by instantiating various diff-dataset generation functions. Depending on whether a binary or a multiclass diff-classifier has to be created, the particular data generation approach generates a diff-dataset without target y for an instance. The diff-dataset target determiner adds the list Y by comparing the predictions of the black boxes to the dataset. Finally, DiRo2C trains, using the resulting diff-dataset Z_{Diff}, either a binary or a multiclass diff-classifier, to predict the differences.

3.1 Data Approaches

DiRo2C tests different data approaches against each other. In detail, it provides global and local approaches (cf. Section 2.4 and Chapter 4). One main difference between

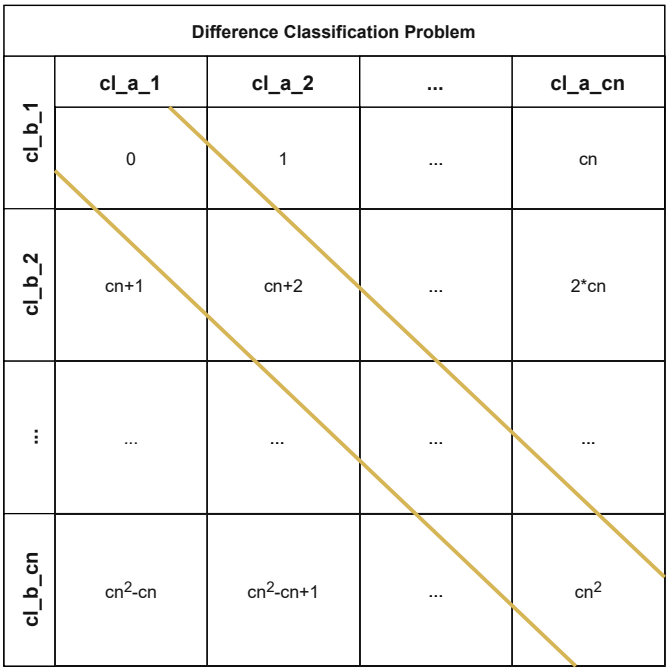


Figure 3.3: Difference classification problem generalized to cn possible classes of the black boxes.

the two principles is that the global data approaches are not depending on a particular instance x. We implement two functions to generate *global synthetic data* and to use already existing *global real data*. The local approaches to return the diff-dataset are the *local genetic neighborhood generation* function and the *local real data* function that depends on a particular instance x. We test against approaches where we use the existing dataset of the black boxes. For the experiments, the black boxes are trained with well-known benchmark datasets. The *global real data* and the *local real data* approach using the already existing instances (of the datasets of the black boxes) to provide real data.

3.1.1 Running Example Dataset

For illustrating the different data approaches, we use the synthetic classification dataset as seen in Figure 3.5 to train black box A. We manipulate this dataset to train black box B. This dataset example can be compared with the initial *difference classification problem* example (see Figure 3.1). The left dataset for black blox A shows a two-dimensional dataset with the continuous features x_1 and x_2. It contains 300 instances (datapoints) with the following properties for feature x_1: $min = -293.39$, $max = 437.35$, $\mu = -2.92$, and $\sigma = 187.69$ and with the following properties for feature x_2: $min = -251.96$, $max = 316.10$, $\mu = -3.23$, and $\sigma = 101.08$. The instances of the datasets are classified into two classes “0” and “1”. 150 instances of the left dataset are assigned to the class “0”, and 150 instances are assigned to the class “1”. The instances of the dataset for black

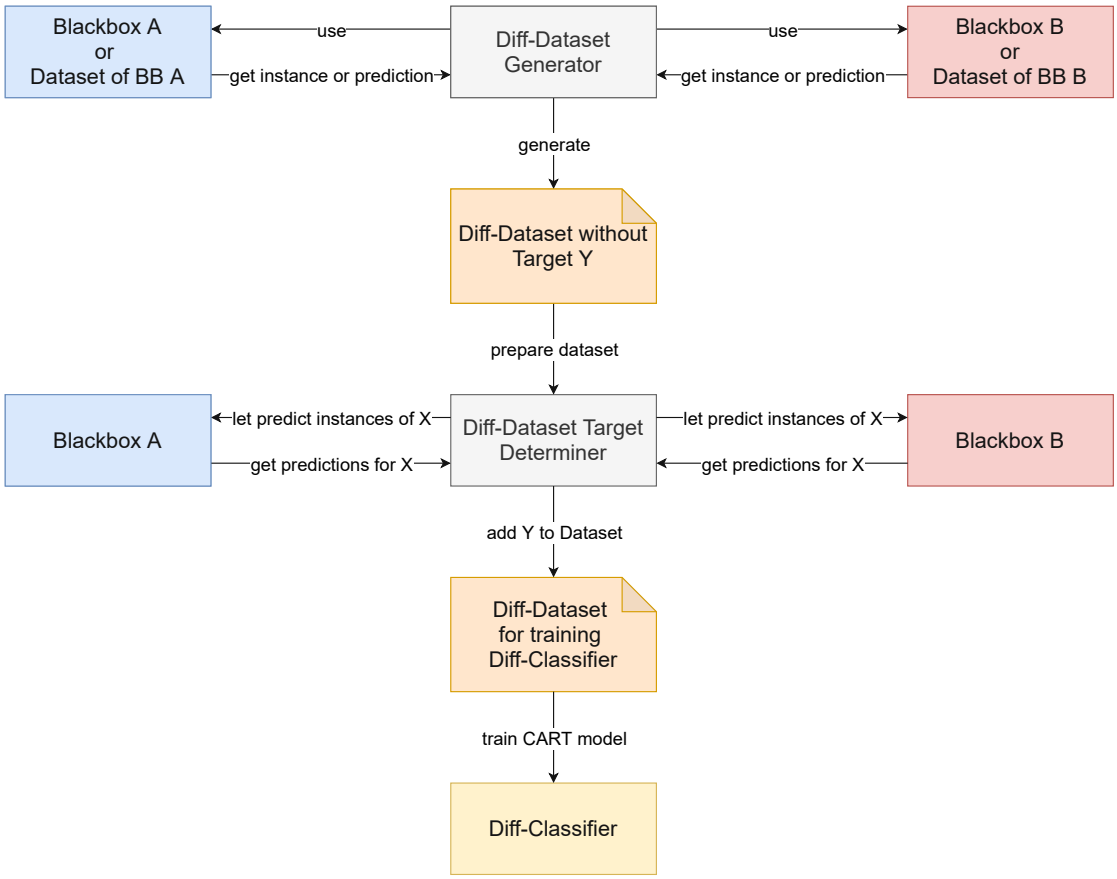


Figure 3.4: Process and involved components for training the Diff-Classifier

box A are generated by the sklearn "make_classification" function with the following parameters: make_classification(n_samples = 300, n_features = 2, n_informative = 1, n_redundant = 0, n_classes = 2, random_state = 2, n_clusters_per_class = 1, class_sep = 1.8, flip_y = 0, scale = 100). The right dataset is manipulated on the basis of the dataset of black box A as follows: ($x_1 <= 0$ and $x_2 <= -100 : y = 0$), ($x_1 > 0$ and $x_2 <= -100 : y = 1$), ($x_1 < 150$ and $x_2 > -100$ and $x_2 < 100 : y = 0$), ($x_1 >= 150$ and $x_2 > -100$ and $x_2 < 100 : y = 1$), ($x_1 = -200$ and $x_2 >= 100 : y = 0$), and ($x_1 > -200$ and $x_2 >= 100 : y = 1$). 165 instances are now assigned to the class "0", and 135 instances are assigned to the class "1". Thus, we only manipulate the outcome classes of the instances of the dataset for black box B. The properties for the feature x_1 and x_2 remain unchanged. We use the datasets[1] for black box A and B in the following chapters as a running example.

The following data examples are based on the presented datasets of black box A and B (see Figure 3.5). We show in the next figures the resulting datasets of the different data

[1]Datasets are available under: https://doi.org/10.5281/zenodo.5325335

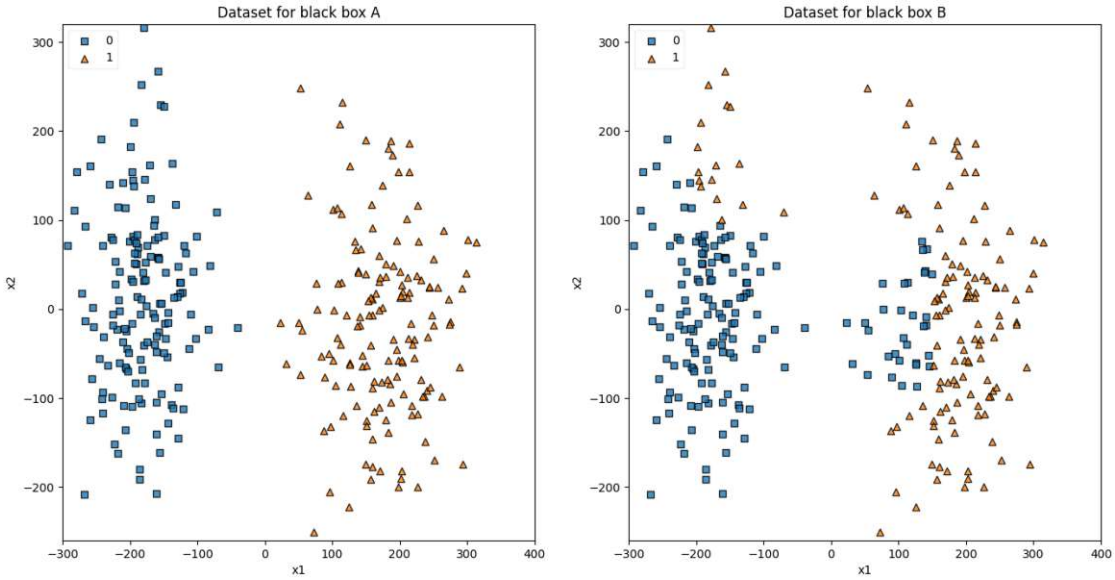


Figure 3.5: Running example of a synthetic classification dataset for black box A and a manipulated dataset for black box B. The plot of the dataset of black box B shows the different shifted boundaries of the class membership.

approaches for training the classifier to recognize differences between the black boxes. So, each instance is predicted by black box A and B, and after that, the predictions are compared to assign the instance to the corresponding difference-class (we explain the target determination in detail in Section 3.2). Therefore, we show the resulting decision boundaries (indicated by the black drawn lines) of the trained black box A and B in Figure 3.6. The both plots show how the instances are differently predicted by black box A and B. In our running example, we split the generated dataset into a training and test dataset with a ratio of 80 to 20 percent. After that, we train black box A and B using the training dataset and a decision-tree based CART optimized algorithm. The plot shows the resulting decision boundaries of black box A and B. The decision boundaries represent the learned conditions (rules) for the prediction. Black box A classifies the instances with the following rule: $(x_1 <= -8.545 : y = 0)$ and $(x_1 > -8.545 : y = 1)$. The right plot shows the three manipulated, different shifted decision boundaries of the trained black box classifier B on the x_1 axis.

To compare the different basis for decision-making of the two black boxes in detail, Figure 3.7a shows the decision tree of black box A and Figure 3.7b shows the decision tree of the manipulated black box B. Figure 3.7b depicts the manipulated decision boundaries for black box B.

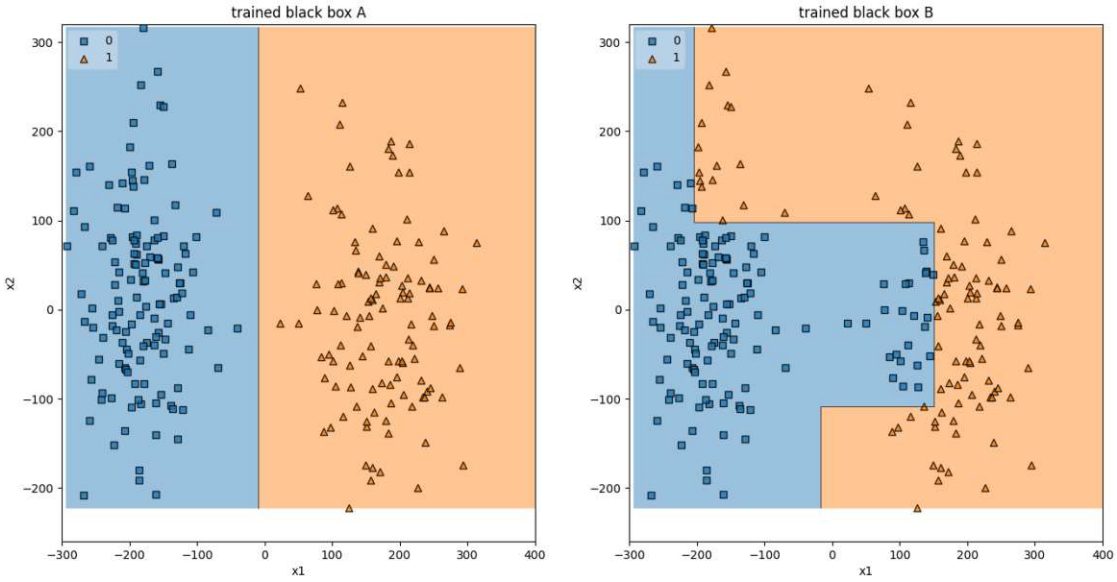
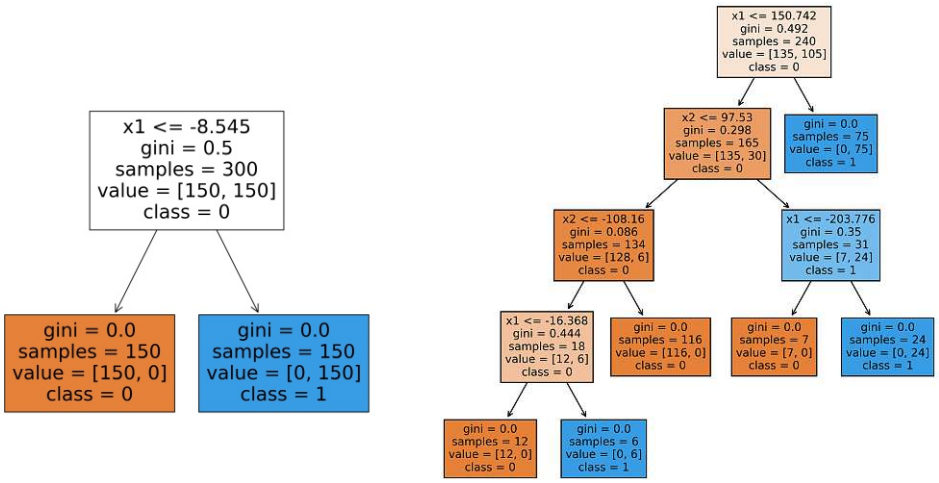


Figure 3.6: Trained black box classifier A and B with the corresponding decision boundaries based on the running example dataset.

3.1.2 Global Synthetic Data

This approach generates the data Z_{Diff} by computing data randomly according to the feature value spectrum. Figure 3.8b shows the generated instances (data points) for the binary (see left plot) and the multiclass diff-classifier (see right plot). The values for the instances are created randomly for a defined number of instances. The approach computes the data based on the concatenated training and test dataset of black box A and B. Therefore, the approach covers the entire feature value spectrum of both black boxes. The datasets for the binary and multiclass classifier are generated independently of each other, which means the instances of the datasets shown in the plots are not identical. The dataset for the binary diff-classifier contains 1,300 instances. 990 instances are assigned to the class "0" ("no diff"), and 310 instances are assigned to the class "1" ("diff"). The dataset for the multiclass diff-classifier also contains 1,300 instances, whereby 571 instances are classified into class "00", 424 instances are classified into class "11", 237 instances are classified into class "10", and 68 instances are classified into class "01". Possible values for the discrete features are determined to produce values for new instances using a random function. In both examples, the continuous features are computed by calculating μ and σ to get a sample from a Gaussian distribution per feature. The dataset 3.8a in Figure contains only continuous features created with $\mu = -2.92$ and $\sigma = 187.69$ for feature x_1 and $\mu = -3.23$, and $\sigma = 101.08$ for feature x_2. The approach generates increased instances in those areas instances of the original datasets also occur more frequently. It is used for the model explanation problem to train an interpretable global predictor.



(a) Decision tree of black box classifier A

(b) Decision tree of manipulated black box classifier B

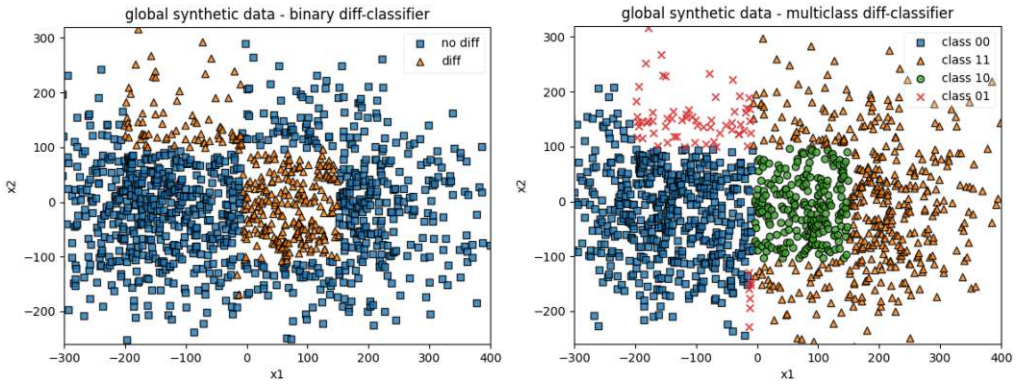
Figure 3.7: Decision tree of trained binary black box classifier A and manipulated binary black box classifier B (Figure 3.5 shows the datasets of the black boxes). If the condition for a particular node is met, the path on the left of the decision tree is chosen.

3.1.3 Global Real Data

The global real data approach uses already existing data. Figure 3.9 illustrates the approach of returning global real data. The figure shows the generated instances (data points) for the binary (see left plot) and the multiclass diff-classifier (see right plot). We are using the concatenated training and test datasets of black box A and B to train the diff-classifier. The datasets for the binary and multiclass classifier are, in that case, identical. The unbalanced dataset for the binary diff-classifier contains 600 instances. 502 instances are assigned to the class “0” (“no diff”), and 98 instances are assigned to the class “1” (“diff”). The also unbalanced dataset for the multiclass diff-classifier contains again 600 instances, whereby 266 instances are classified into class “00”, 236 instances are classified into class “11”, 64 instances are classified into class “10”, and 34 instances are classified into class “01”. Since the approach only uses data from the original datasets of the black box models, it can happen, as in this example, that the class distribution is unbalanced. The figures also show that the instances do not cover the entire data space. Therefore, the coverage of the data space depends on the original data. This approach is also used for the model explanation problem to train an interpretable global predictor.

3.1.4 Local Genetic Neighborhood Data

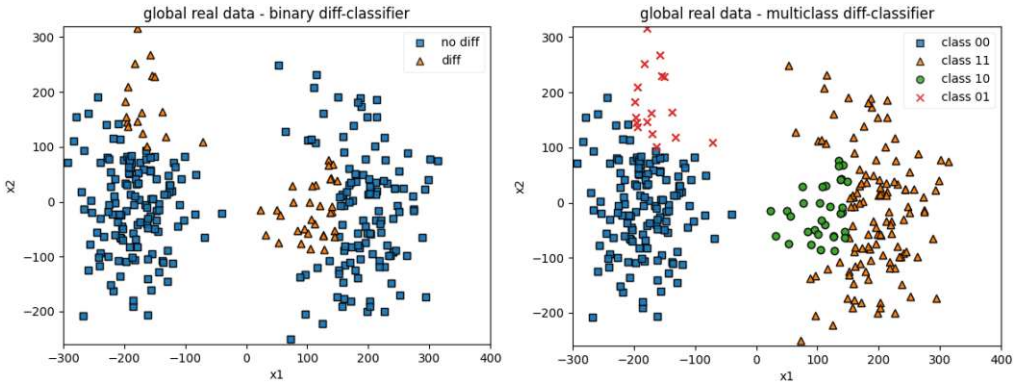
The local genetic neighborhood data generation approach is originally based on the work of Guidotti et al. [GMR+18]. This approach uses a genetic algorithm implementation with biological evolution-inspired principles. We use this approach to generate depending



(a) Dataset for training the binary diff-classifier

(b) Dataset for training the multiclass diff-classifier

Figure 3.8: Generated global synthetic datasets for training a binary and a multiclass diff-classifier to detect differences between black box A and B (Figure 3.5 shows the datasets of the black boxes). The approach generates the data independent of a particular instance x.



(a) Dataset for training the binary diff-classifier

(b) Dataset for training the multiclass diff-classifier

Figure 3.9: Global real datasets for training a binary and a multiclass diff-classifier to detect differences between black box A and B (Figure 3.5 shows the datasets of the black boxes).

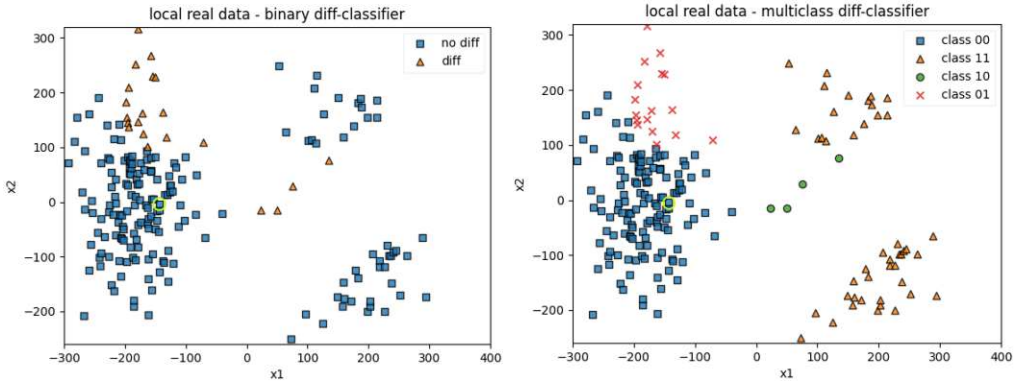
on a particular instance x, and the predictions of the black box models new instances. The new instances are generated in the local vicinity of x depending on a fitness function. According to this function the fittest instances are selected. The aim is to generate increased instances (data points) locally as close as possible to the instance x to recognize the differences between the black boxes. Since this data generation method is a crucial component of DiRo2C, a separate chapter is dedicated to this data approach. In Chapter 4, the original approach of LORE and the adaptation for DiRo2C are explained in detail.

3.1.5 Local Real Data

The local real data selecting approach returns Z_{Diff}, which contains a predefined number l of instances from the concatenated dataset of the black box models (see Figure 3.10). The figure shows the selected instances (data points) for the binary (see left plot) and the multiclass diff-classifier (see right plot). We are using the concatenated training and test datasets of black box A and B to train the diff-classifier. The datasets for the binary and multiclass classifier are in that case identical. The unbalanced dataset for the binary diff-classifier contains 200 instances. 179 instances are assigned to the class "0" ("no diff"), and 21 instances are assigned to the class "1" ("diff"). The also unbalanced dataset for the multiclass diff-classifier also contains 200 instances, whereby 133 instances are classified into class "00", 46 instances are classified into class "11", 4 instances are classified into class "10", and 17 instances are classified into class "01". This approach uses the same distance function as the *local genetic neighborhood generation* approach for computing the local closest instances compared to the to be recognized instance x (marked with a yellow circle in the figure and located at: $x_1 = -143.91$ and $x_2 = -4.53$). Depending on the computed distance (see Equation 3.1) the first l instances are selected. In this example, l is deliberately chosen to be small to show the difference between the global real data approach (cf. 3.1.3). The distance ($SimpleMatch$) for the categorical (discrete) features is determined by counting the matching categorical features. Whereby h denotes the number of discrete and $m - h$ the number of continuous features (with m designating the total number of input features). The sum of the matching features is weighted by the total number of categorical features. The distance for the continuous features is calculated by using the normalized Euclidean distance ($NormEuclid$). Finally, d returns the weighted sum of $SimpleMatch$ and $NormEuclid$.

$$d(x, z) = \frac{h}{m} \cdot SimpleMatch(x, z) + \frac{m - h}{m} \cdot NormEuclid(x, z) \tag{3.1}$$

According to that distance function, the local real data approach computes for every instance the distance to the to be detected instance x. After that, the instances are sorted by distance, and the first k instances are selected. The approach selects k instances twice. Once dependent on black box A and once dependent on black box B. The approach also divides the instances into two different groups. One group contains instances that are classified into the class with $y = 0$ by the black box. The other group contains instances that are classified into the class with $y = 1$ by the black box. These two groups are also sorted by distance. Then, the approach checks the already selected instances whether there is at least a ratio of 70 to 30 percent between the two mentioned groups. If the condition is not met, the approach tries to select instances from the respective disadvantaged group instead to establish the desired ratio. The selected instances for black box A and B are then merged and checked for uniqueness.



(a) Dataset for training the binary diff-classifier

(b) Dataset for training the multiclass diff-classifier

Figure 3.10: Selected local real data for training a binary and a multiclass diff-classifier to detect differences between black box A and B (Figure 3.5 shows the datasets of the black boxes). Given on the yellow-marked instance (Seed 1) the local real dataset approach selects the instances from the concatenated datasets of the black box models.

3.2 Target Determination

The target determining component has to compute the target Y_{Diff} to provide a diff-dataset for training a binary or a multiclass classifier. The *BuildDiffDataset* Algorithm 3.1 computes the outcomes of X_{Diff} for black box A and B, compares the results and returns the determined label. Depending on the passed comparing function, a diff-dataset for a binary (see Algorithm 3.3) or a multiclass (see Algorithm 3.2) diff-classifier is returned. At first, the Algorithm 3.1 obtains the labels Y_A from black box A: $b_A(X_{Diff})$=Y_A (see line number 1) and the targets Y_B for black box B: $b_B(X_{Diff})$=Y_B (see line number 2). After that, in line number 3, the *Compare* function computes the label for the diff-dataset Y_{Diff} as detailed below. Finally in line number 4, the algorithm concatenates the resulting Y_{Diff} with the X_{Diff} along the horizontal axis to return the determined diff-dataset with the outcome class for recognizing the differences between black box A and B in line number 5.
As already mentioned, DiRo2C provides a binary and a multiclass diff-classifier. Algorithm 3.2 shows how the *CompareBinary* function works to provide Y_{Diff} for the binary diff-classifier. In that case it compares the predicted outcomes of black box A (Y_A) and B (Y_B) and inverts the result so that the diff-classifier predicts the binary class 1 if black box A and black box B predict different outcomes (see line number 1). Since we are using metrics like *Accuracy* and F_1-*score* to measure how accurately the diff-classifier predicts the differences, we have to dedicate the "diff" outcome class as the positive class 1. Algorithm 3.3 depicts the comparing function to return Y_{Diff} to train a multiclass classifier for predicting every different combination of predictions for black boxes A and B. Therefore, the algorithm iterates over the predictions of black box A (Y_A) and B

Algorithm 3.1: $BuildDiffDataset(b_A, b_B, X_{Diff}, Compare)$

Input: b_A - black box A, b_B - black box B, X_{Diff} - diff-dataset without Y, $Compare$ - compare function to determine Y
Output: YX_{Diff} - determined diff-dataset

1 $Y_A \leftarrow b_A.predict(X_{Diff})$; // predict Y of black box A
2 $Y_B \leftarrow b_B.predict(X_{Diff})$; // predict Y of black box B
3 $Y_{Diff} \leftarrow Compare(Y_A, Y_B)$; // determine Y for diff-dataset
4 $YX_{Diff} \leftarrow Y_{Diff} \oplus X_{Diff}$; // concatenate Y_{Diff} and X_{Diff}
5 **return** YX_{Diff};

Algorithm 3.2: $CompareBinary(Y_A, Y_B)$

Input: Y_A - predictions of black box A, Y_B - predictions of black box B
Output: Y_{Diff} - determined target for a binary diff-dataset

1 $Y_{Diff} \leftarrow Invert(Y_A == Y_B)$; // determined Y_{Diff}
2 **return** Y_{Diff};

(Y_B) (see line number 3). Then, it compares for every instance the outcome of black box A (Y_{Ai}) and B (Y_{Bi}). If Y_{Ai} predicts 0 and Y_{Bi} 0 then, it appends class "a" for the combination "00" to Y_{Diff} (see line number 4 to 6). Similiary, we define for the combination "11" the class "b" (see line number 7 to 9), for the combination "10" the class "c" (see line number 10 to 12), and for the combination "01" the class "d" (see line number 13 to 15). After the iteration in line number 17, the $CompareMulticlass$ returns Y_{Diff} to train the multiclass diff-classifier.

3.3 Summary

In this chapter, we have presented the approach of DiRo2C to detect differences between binary black box classifiers and the main components. We have explained the *difference explanation problem*, which we have generalized to any number of difference-classes, and the term *difference-classes*. We also have pointed out that we are focusing exclusively on designing and implementing a solution for the 2 (binary) and 4 (multiclass) *difference-class* problem. We have introduced our running example, which we are using through our entire work and the different local and global (difference) data approaches. We have explained in this chapter the *global synthetic data*, the *global real data*, and the *local real data* approach. In the next chapter, we will explain the *local genetic neighborhood* generation approach in detail. It is noteworthy that a general advantage of the synthetic data generation approaches is that any number of instances can be generated synthetically by choice. This approach enables independence from the original data and the class distribution. In addition, the multiclass diff-classifier has the advantage of classifying all possible different outcome class combinations of two binary black box classifiers.

Algorithm 3.3: $CompareMulticlass(Y_A, Y_B)$

Input: Y_A - predictions of black box A, Y_B - predictions of black box B
Output: Y_{Diff} - determined target for a multiclass diff-dataset

```
1  Y_Diff ← ([ ]);                                   // init Y_Diff
2  N ← length(Y_A);                  // determine length of list Y_A
3  for i ← 0 to N − 1 do
       // iterate over list Y_A and Y_B
4      if Y_Ai == 0 And Y_Bi == 0 then
           // append class 'a' for combination '00' to Y_Diff
5          Y_Diff ← Y_Diff + 'a';
6      end
7      if Y_Ai == 1 And Y_Bi == 1 then
           // append class 'b' for combination '11' to Y_Diff
8          Y_Diff ← Y_Diff + 'b';
9      end
10     if Y_Ai == 1 And Y_Bi == 0 then
           // append class 'c' for combination '10' to Y_Diff
11         Y_Diff ← Y_Diff + 'c';
12     end
13     if Y_Ai == 0 And Y_Bi == 1 then
           // append class 'd' for combination '01' to Y_Diff
14         Y_Diff ← Y_Diff + 'd';
15     end
16 end
17 return Y_Diff;
```

CHAPTER 4

Simplistic Genetic Neighborhood

In this chapter, we focus on the local genetic neighborhood data generation approach. First, we explain the basic principles of the LORE approach. Then, we describe how we enable the original approach for our simplistic approach adapted to generate a dataset depending on an instance x to recognize differences between two black boxes. This dataset is used to train a local classifier to recognize local differences close to an instance x between the black box models.

4.1 LORE's Genetic Instance Generation Approach

The promising approach to generate the dataset to train the diff-classifier for DiRo2C is the genetic neighborhood generation of LORE [GMR+18]. The approach of Guidotti et al. is promising because it can generate synthetic instances in the local vicinity close to the to be explained instance x. That leads to a higher data density close to x, as the experiments of Guidotti et al. show. By applying the approach, we try to predict the local differences accurately. The original approach (see Algorithm 4.1) aims to find and create a set Z_x (see line number 4), with feature values and characteristics similar to the instance x to approximate the local behavior of a black box model. Therefore, the objective is to create a set of Z that includes instances with both outcome values of the black box: $Z = Z_{=} \cup Z_{\neq}$ (see line numbers 2 to 4). Whereby, the instances $z \in Z_{=}$ have to meet the condition $b(z) = b(x)$, and the instances $z \in Z_{\neq}$ the condition $b(z) \neq b(x)$ (i.e., instances where the black boxes return the same predicted outcomes as for the instance to be explained, and instances where the predicted outcomes differ). The great advantage of the LORE approach is that the instance generation is independent of an available training dataset. That means it is not necessary to rely on the dataset of the black box. Guidotti et al. describe in their work [GMR+18] that their approach for the instance selection is similar to active learning [FZL12] and also contains evolutionary approaches [DGH10].

Algorithm 4.1: $LORE(x, b, N)$, Source: [GMR+18]

Input: x - instance to explain, b - black box, N - # of neighbors
Output: e - explanation of x

```
1 G ← 10; pc ← 0.5; pm ← 0.2;                          // init parameters
2 Z_= ← GeneticNeigh(x, fitness^x_=, b, N/2, G, pc, pm);   // generate neigh
3 Z_≠ ← GeneticNeigh(x, fitness^x_≠, b, N/2, G, pc, pm);   // generate neigh
4 Z ← Z_= ∪ Z_≠;                                       // merge neighborhoods
5 c ← BuildTree(Z);                                    // build decision tree
6 r = (p → y) ← ExtractRule(c, x);                     // extract decision rule
7 Φ ← ExtractCounterfactuals(c, r, x);                 // extract counterfactuals
8 return e = ⟨ r, Φ ⟩;
```

LORE creates the balanced instances $z \in Z_= \cup Z_{\neq}$ by adapting a genetic algorithm approach and maximizing the following fitness functions (see Equation 4.1 and 4.2):

$$fitness^x_=(z) = I_{b(x)=b(z)} + (1 - d(x, z)) - I_{x=z} \tag{4.1}$$

$$fitness^x_{\neq}(z) = I_{b(x)\neq b(z)} + (1 - d(x, z)) - I_{x=z} \tag{4.2}$$

where d is a distance function (see Equation 3.1): $d : \mathcal{X}^{(m)} \to [0, 1]$ and $I = 1$, if the condition is *true*, else $I = 0$. In both cases, the fitness functions search for instances where the feature characteristics are close to x: $(1 - d(x, z))$ but not equal to x (see term $I_{x=z}$). The fitness function defined in Equation 4.1 looks especially for instances where the black box b returns for a potential instance z_0 the same outcome as for x. In contrast, the function defined in Equation 4.2 guarantees the generation of instances for which b predicts a different outcome. Thus, the function $fitness^x_=(z_0)$ for an instance z_0 where $b(x) \neq b(z_0)$ and $x \neq z_0$ leads to a result smaller than 1. Instead, $b(x) = b(z_0)$ leads to a result greater or equal than 1. For the instance x itself it returns 1, $fitness^x_=(x) = 1$. Therefore, the function guarantees the generation of instances that are different from x and are as close as possible to x. Another crucial element is the already explained distance function (see Equation 3.1) of the fitness function. Both fitness functions are passed to Algorithm 4.2 to generate the neighbors.

The LORE method generates the neighbor instances of x through an instance of the evolutionary approach (see Algorithm 4.2, which is described in the work of Bäck et al. [BFM00]). Figure 2.8 shows an example of the genetic neighborhood generation for an instance x (cf. Section 2.6). It is a particular instantiation of generational genetic algorithms for evolutionary prototype generation [DGH10], whereby prototypes are generally an optimized subset of a dataset. But in the context of LORE, the generation approach is adapted to generate new instances. Guidotti et al. mention that the following works: [Bal94, CHL05, Esh91, WO06] propose an approach to use genetic algorithms

Algorithm 4.2: $GeneticNeigh(x, fitness, b, N, G, pc, pm)$, Source: [GMR+18]

Input: x - instance to explain, b - black box, $fitness$ - fitness function, N - population size, G - # of generations, pc - crossover probability, pm - mutation probability

Output: Z - neighbors of x

1 $P_0 \leftarrow \{x|\forall 1...N\}$; $i \leftarrow 0$; // population init
2 $evaluate(P_0, fitness, b)$; // evaluate population
3 **while** $i < G$ **do**
4 $\quad P_{i+1} \leftarrow select(P_i)$; // select sub-population
5 $\quad P'_{i+1} \leftarrow crossover(P_{i+1}, pc)$; // mix records
6 $\quad P''_{i+1} \leftarrow mutate(P'_{i+1}, pm)$; // perform mutations
7 $\quad evaluate(P''_{i+1}, fitness, b)$; // evaluate population
8 $\quad P_{i+1}$=P''_{i+1}; // update population
9 $\quad i \leftarrow i + 1$;
10 **end**
11 $Z \leftarrow P_i$;
12 **return** Z;

by integrating classifiers decisions within the fitness function. Genetic algorithms were proposed by J. H. Holland and D. E. Goldberg and are based on the biological evolution process [Gol89, Hol92]. Besides, natural selection, the concept of survival of the fittest, and genetics inspire those algorithms. Genetics algorithms have the following three essential elements in common [Gol89, GMR+18, Hol92, KCK20]: *genetic representations*, *fitness selection*, and biological inspired *operators*.

The genetic representations, which are generally called chromosomes, are depending on the solution domain to solve a particular problem. In the specific case of LORE, the chromosomes correspond to instances in the feature space $\mathcal{X}^{(m)}$, which supports variation and selection operations.

The fitness function identifies the best chromosomes regarding the specific case (cf. 4.1 and 4.2). The fittest chromosomes are more likely to survive and reproduce. These, therefore, have an enormous influence on the next generation and their properties.

The genetic neighborhood generation approach uses crossover (also called mating) and mutation operators to define the next generation of chromosomes. It includes a stopping criterion that delivers the fittest generation as a result.

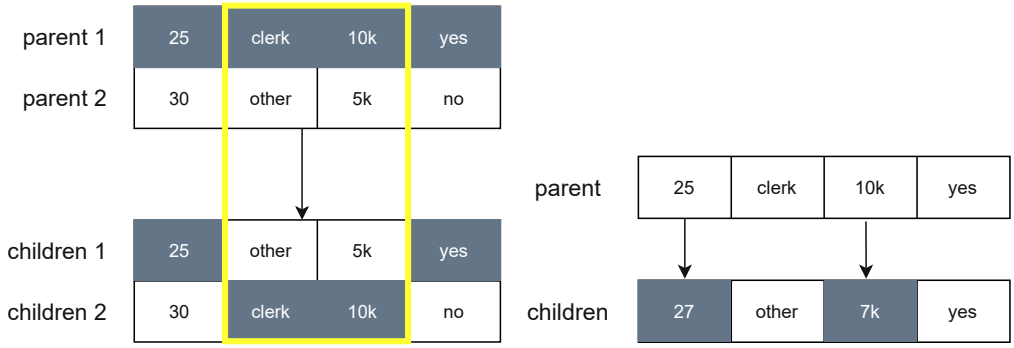
Algorithm 4.2 starts by creating N copies of the to be explained instance x (see line number 1) and initially evaluates the first population P_0 using the fitness functions $fitness^x_=(z)$ (see Equation 4.1) and $fitness^x_{\neq}(z)$ (see Equation 4.2), and the black box b (see line number 2). Then the evolution loop (see line number 3) begins to select the P_{i+1} population (see line number 4). The method *select* (see line number 4) selects a sub-population depending on the computed fitness score for each generated instance (instances with a higher calculated fitness score are more likely to survive). Thus, the *select* method returns a sub-population with the highest fitness score. We want to

mention that the first time the evolution loop is run, all instances of the population would be identical, and thus all instances would have the same fitness score. Guidotti et al. use the "eaSimple" algorithm of the deap library [FDRG+12] to generate genetic instances. In the documentation[1] of the deap algorithms the following is stated: "[...] First, it evaluates the individuals with an invalid fitness. Second, it enters the generational loop where the selection procedure is applied to entirely replace the parental population. The 1:1 replacement ratio of this algorithm requires the selection procedure to be stochastic and to select multiple times the same individual [...]". Therefore, we assume that the parental population is exchanged through a stochastic process before the first evaluation is processed. Thereafter, the crossover operator (to mix the instances) is carried out and places the updated population in P'_{i+1} (see line number 5). Then, the mutation operator is carried out, and the again updated population is placed in P''_{i+1} (see line number 6). A "crossover-probability" pc and a "mutation-probability" pm control the proportion of how many instances are affected by the operations. LORE uses a two-point crossover that randomly picks two crossover features from two selected parents' chromosomes. After that, the crossover features of the parents are swapped (see Figure 4.1a). LORE also applies the mutation operator that randomly replaces the value of a particular feature by using the empirical distribution in case of a continuous feature and by choosing a possible value in case of a discrete feature (see Figure 4.1b). Therefore LORE uses the test dataset of the black box to determine the feature values. Guidotti et al. claim that they use the test dataset of the black box to derive the distribution for experimental purposes [GMR+18]. The resulting population P''_{i+1} is then evaluated by applying the fitness function and the predictions of the black box (see line number 8). The evaluation method provides the fittest individuals, and the loop continues. The goal of the evolution loop is to create the best fitting individuals by terminating after G generations (see line number 3) and to return the population P_G as Z (see line number 10). The algorithm 4.2 runs, as already mentioned, for both fitness functions: $fitness^x_=$ and $fitness^x_{\neq}$ to create the set $Z = Z_= \cup Z_{\neq}$.

4.2 Adaptions for DiRo2C

For DiRo2C, we adapt the LORE Algorithm 4.1. As in the original approach, the algorithm generates a neighborhood depending on the instance x to explain. Algorithm 4.3 shows our adaption. In this approach, we use the unchanged genetic algorithm of LORE (see Algorithm 4.2) to create Z. The difference, however, is that we create Z for both black boxes A and B (see line numbers 2 to 6). Then, we merge the created instances Z_A and Z_B: $Z = Z_A \cup Z_B$ (see line number 7). Z and the instances it contains are checked for uniqueness and the redundant instances are removed. After that, we use Z and black box A and B to determine the target Y (cf. Section 3.2) for the diff-dataset (see line number 8). DiRo2C uses the already presented $BuildDiffDataset$ function (see Algorithm 3.1) for building the diff-dataset (see line number 8). After that, the method

[1] The documentation of the deap algorithms is available under: `https://deap.readthedocs.io/en/master/api/algo.html`



(a) Example of a crossover operation (b) Example of a mutation operation

Figure 4.1: Examples of a crossover and a mutation operation. Examples used from: [GMR+18]

Algorithm 4.3: *DiRo2C_GN*(x, b_A, b_B, N), Adapted from: [GMR+18]

Input: x - instance to explain, b_A - black box A, b_B - black box B, N - # of neighbors

Output: dc - diff-classifier, Z_{Diff} - merged neighborhood of black box A and B and determined target Y

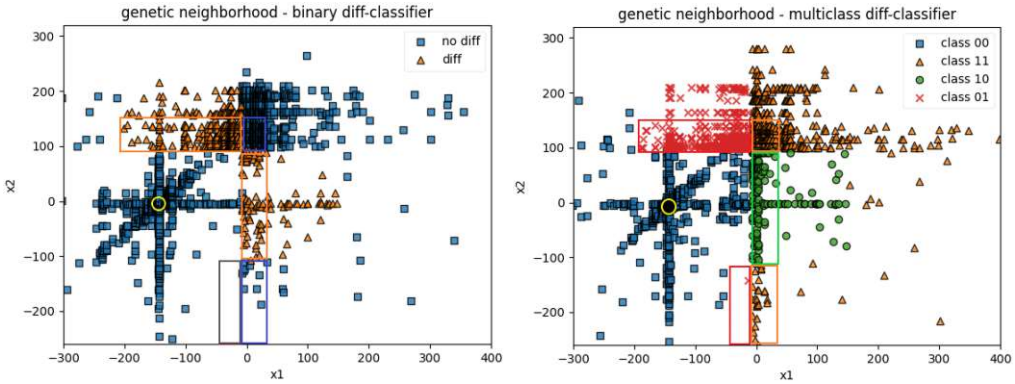
1 $G \leftarrow 10$; $pc \leftarrow 0.5$; $pm \leftarrow 0.2$; // init parameters
// generate neighbors for black box A
2 $Z_{A=} \leftarrow GeneticNeigh(x, fitness^x_=, b_A, N/2, G, pc, pm)$;
3 $Z_{A\neq} \leftarrow GeneticNeigh(x, fitness^x_{\neq}, b_A, N/2, G, pc, pm)$;
4 $Z_A \leftarrow Z_{A=} \cup Z_{A\neq}$; // merge neighborhoods
// generate neighbors for black box B
5 $Z_{B=} \leftarrow GeneticNeigh(x, fitness^x_=, b_B, N/2, G, pc, pm)$;
6 $Z_{B\neq} \leftarrow GeneticNeigh(x, fitness^x_{\neq}, b_B, N/2, G, pc, pm)$;
7 $Z_B \leftarrow Z_{B=} \cup Z_{B\neq}$; // merge neighborhoods
8 $Z \leftarrow Z_A \cup Z_B$; // merge neighborhoods
9 $Z_{Diff} \leftarrow BuildDiffDataset(b_A, b_B, Z)$; // determine target Y
10 $dc \leftarrow TrainDecisionTreeClassifier(Z_{Diff})$; // train diff-classifier
11 **return** dc, Z_{Diff};

trains the diff-classifier based on the diff-dataset Z_{Diff} (see line number 9). Finally, it returns the dataset Z_{Diff} and the decision tree classifier to detect the differences between black boxes A and B. We want to mention, that for determining the feature value ranges to compute new feature values, we use the concatenated test dataset of black box A and B.

4.3 Illustration of the Generated Datasets

Now, we show generated neighborhood datasets based on the running dataset example that depends on different located instances (seeds).

Seed 1: Figure 4.2a and 4.2b illustrate an example of the neighborhood generation adapted to the problem to recognize local differences close to an instance x between two binary black boxes. For this example, we use the already presented running example classification datasets of black box A and black box B (Figure 3.5 shows the datasets of the black box classifiers). The datasets for the binary and multiclass classifier are generated independently of each other, which means the instances (data points) of the datasets shown in the plots are not identical. The unbalanced dataset for the binary diff-classifier contains 2,246 instances. 1,769 instances are assigned to the class "0" ("no diff"), and 477 instances are assigned to the class "1" ("diff"). The unbalanced dataset for the multiclass diff-classifier contains 2,004 instances, whereby 815 instances are classified into class "00", 615 instances are classified into class "11", 196 instances are classified into class "10", and 378 instances are classified into class "01". Each dataset is generated based on the instance x highlighted in yellow and both are located at: $x_1 = -143.91$ and $x_2 = -4.53$. Based on x, synthetic instances for both black boxes are generated independently. By applying the genetic algorithm to generate a neighborhood for black box A and B, the aim is to generate increased instances close to the yellow marked instance x to recognize the local boundaries where the instances are classified into different difference-classes. Using Figure 4.2a, we want to explain in which areas more instances and in which areas fewer instances are generated to train a binary diff-classifier. The marked rectangles show areas where the instances are assigned into the corresponding binary difference-class and the changes of the classification. In all marked areas (except in the bottom blue and dark grey area), which are close to x, increased instances are generated. The increased instances in that areas help to differentiate between the various difference-class areas. But it is noticeable that the dark gray contains no instances. It can also be seen that fewer instances are generated in the areas that are further away from x. Due to the distance function (see Equation 3.1), which is part of the fitness function, instances that are far away are rated worse and therefore have a lower probability of being selected based on the genetic algorithm. In contrast, using Figure 4.2b, we want to explain in which areas more instances and in which areas fewer instances are generated to train a multiclass diff-classifier. The marked rectangles show areas where the instances are differently classified into the multiclass difference-classes. Again, in all marked areas (except in the bottom orange area and the bottom red area), which are close to x, increased instances are generated. It is noticeable that the bottom red area contains only one instance. We want to explain why more instances are generated in the mentioned areas: Black Box A and B generate new instances independently of each other. Because of both black boxes the approach generates increased instances close to x (see area close to the yellow marked instance) due to the fitness function $fitness^{x}_{=}$. The approach generates in this area also increased instances because both black boxes A and B predict the same result



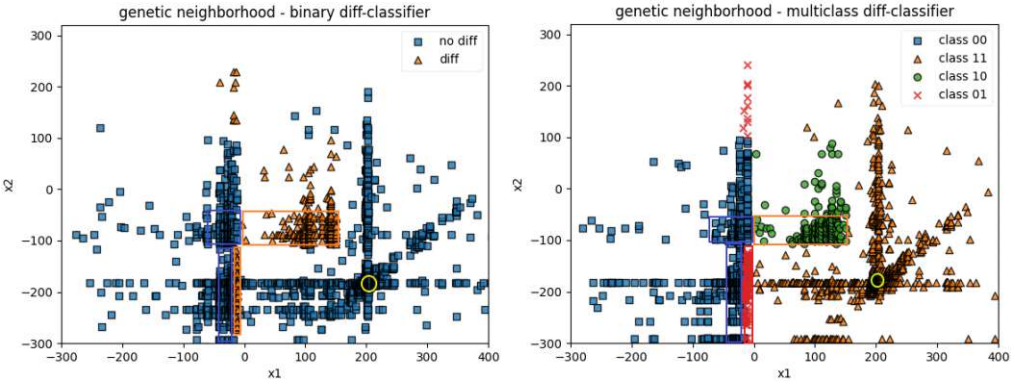
(a) Dataset for training the binary diff-classifier

(b) Dataset for training the multiclass diff-classifier

Figure 4.2: Generated neighborhood datasets with the initial located instance to be recognized (Seed 1) for training a binary and a multiclass diff-classifier to detect local differences close to instance x between black box A and B (Figure 3.5 shows the datasets of the black boxes). Depending on the yellow-marked instance, the proposed approach (see Algorithm 4.3) of DiRo2C generates the shown dataset.

in this area close to x. Due to the fitness function $fitness^x_{\neq}$, and because of black box B, the approach generates increased instances in the upper green (see binary dataset, Figure 4.2a) and upper red area (see multiclass dataset, Figure 4.2b). Due to the fitness function $fitness^x_{\neq}$, and because of both black boxes, the approach generates increased instances in the upper blue (see binary dataset) and upper green area (see multiclass dataset). Due to the fitness function $fitness^x_{\neq}$, and because of black box A, the approach generates increased instances in the lower green (see binary dataset) and orange area (see multiclass dataset). Because of both black boxes and due to the fitness function $fitness^x_{\neq}$ it also generates instances in the lower blue (see binary dataset) and lower green area (see multiclass) but with noticeably fewer generated instances. The dark grey (see binary dataset) and the lower red area (see multiclass dataset) are only influenced by black box B and those areas are neglected by the genetic data generation approach. As explained, due to the distance function, those areas further away from instance x are more disadvantaged.

Seed 2: Now, we show a second example, where we choose the to be recognized instance x differently. The datasets shown in Figure 4.3 are based on the yellow marked instances x. Both instances are located at: $x_1 = 203.73$ and $x_2 = -182.83$. The datasets for the binary and multiclass classifier are generated independently of each other, which means the instances (data points) of the datasets shown in the plots are not identical. The unbalanced dataset for the binary diff-classifier contains 1,951 instances. 1.611 instances are assigned to the class “0” (“no diff”), and 340 instances are assigned to the class “1” (“diff”). The unbalanced dataset for the multiclass diff-classifier contains 1,825 instances,



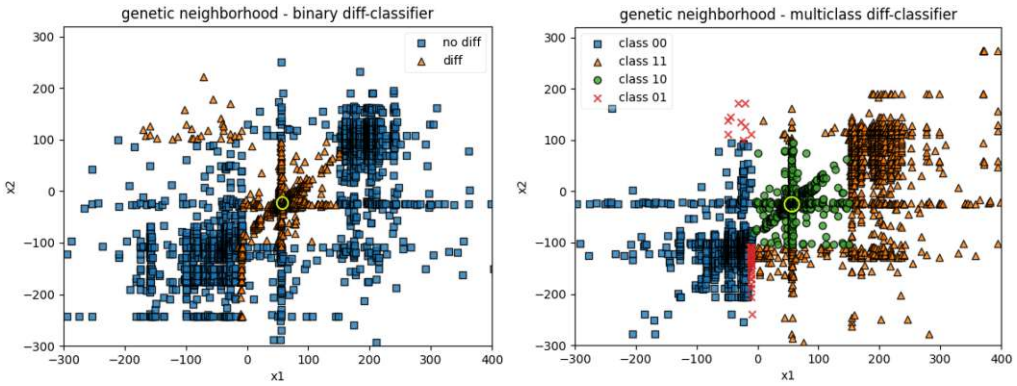
(a) Dataset for training the binary diff-classifier

(b) Dataset for training the multiclass diff-classifier

Figure 4.3: Generated neighborhood datasets with the instance to be recognized in the right bottom corner (Seed 2) for training a binary and a multiclass diff-classifier to detect local differences close to instance x between black box A and B (Figure 3.5 shows the datasets of the black boxes). Depending on the yellow-marked instance, the proposed approach (see Algorithm 4.3) of DiRo2C generates the shown dataset.

whereby 543 instances are classified into class "00", 968 instances are classified into class "11", 220 instances are classified into class "10", and 94 instances are classified into class "01". Using Figure 4.3a, we want to explain in which areas more instances and in which areas fewer instances are generated to train a binary diff-classifier. Again we use rectangles to mark areas where the instances are assigned into the corresponding binary difference-class and the changes of the classification. In all marked areas, which are close to x, increased instances are generated. In that case, the data generation approach covers locally close to x all various difference-classes with increased generated instances. But it is noticeable that significantly fewer instances are existing in the upper area of the plot. Figure 4.3b shows that the data generation approach again generates increased instances in all marked areas. But it is also noticeable that significantly fewer instances are existing in the upper area of the plot. Also, that relatively few instances are generated in the area of the lower center (binary setting: between class "1" and "0" and multiclass setting: between class "10" and "11").

Seed 3: The datasets of the third example shown in Figure 4.4 are based on the yellow marked instances x. Both are located at: $x_1 = 55.61$ and $x_2 = -24.24$. The datasets for the binary and multiclass classifier are generated independently of each other. That means the instances (data points) of the datasets shown in the plots are not identical. The unbalanced dataset for the binary diff-classifier contains 2,674 instances. 1,733 instances are assigned to the class "0" ("no diff"), and 901 instances are assigned to the class "1" ("diff"). The unbalanced dataset for the multiclass diff-classifier contains 2,833 instances, whereby 827 instances are classified into class "00", 1,164 instances are



(a) Dataset for training the binary diff-classifier

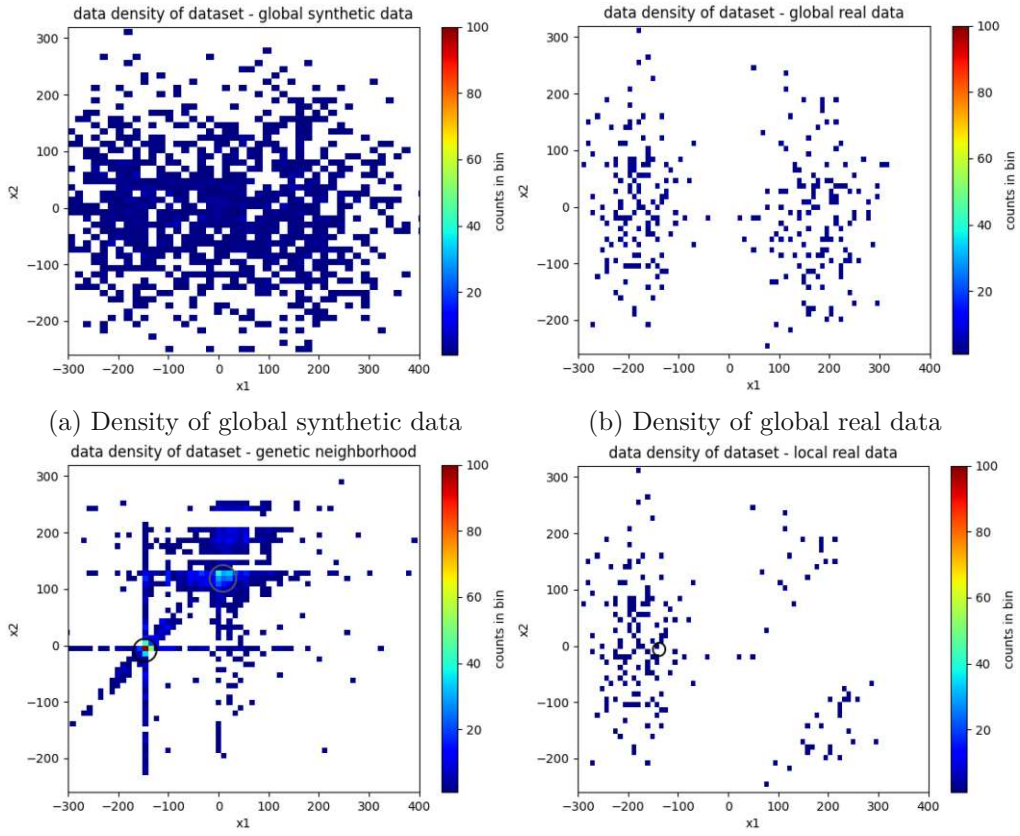
(b) Dataset for training the multiclass diff-classifier

Figure 4.4: Generated neighborhood dataset with the instance to be recognized in the center (Seed 3) for training a binary and a multiclass diff-classifier to detect local differences close to that instance between black box A and B (Figure 3.5 shows the datasets of the black boxes). Depending on the yellow-marked instance, the proposed approach (see Algorithm 4.3) of DiRo2C generates the shown dataset.

classified into class "11", 804 instances are classified into class "10", and 38 instances are classified into class "01". Figure 4.4a again shows that in the various difference-class areas (except in the top "diff"-class area), close to instance x increased instances are generated. Figure 4.4b also shows that the data generation approach generates increased instances in various difference-class areas (except in the top "01"-class area). But it is also noticeable that significantly fewer instances are existing in the upper area of the plot.

4.4 Comparison of the Data Density

This section focuses on the comparison of the data density between the various data approaches. Therefore, we compare the already presented datasets of the different approaches in Figure 4.5. The instance x to be recognized of the local approaches is highlighted by the black circle. We use for the data density plots bins to group the instances on the axes. For the x_1-axis, we use 80 bins, and for the x_2-axis, we use 60 bins. The data density is scaled from 0 to 100 and represents the counts of instances per bin. The white areas show bins where no instances exist. The dataset of the global synthetic data approach shows increased data density in areas around the center. It grounds on the fact that the continuous features in this approach are generated by a Gaussian distribution with the following values for the parameters: $\mu = -2.92$ and $\sigma = 187.69$ for feature x_1 and $\mu = -3.23$, and $\sigma = 101.08$ for feature x_2. (cf. Section 3.1.2). We want to mention that the real data approaches that use the original dataset from the black box models do not generate an increased data density. The global real approach



(a) Density of global synthetic data

(b) Density of global real data

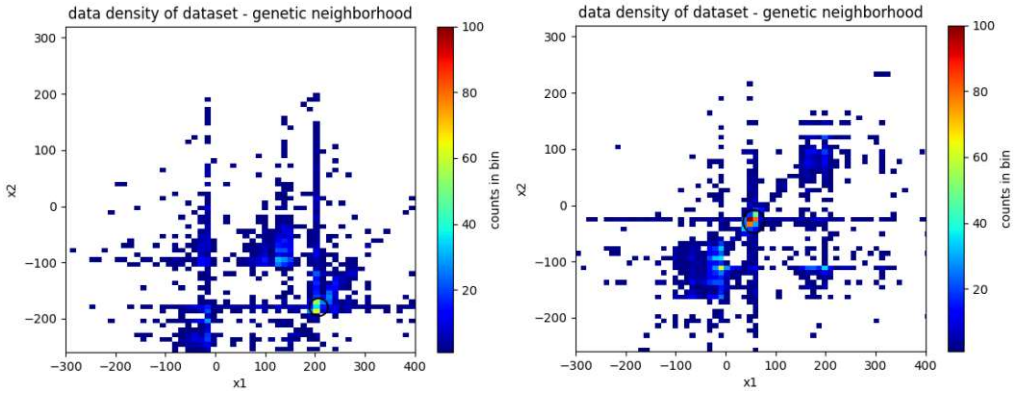
(c) Density of the local genetic neighborhood

(d) Density of local real data

Figure 4.5: Data density of the different local and global data approaches (Seed 1) (Figure 3.5 shows the datasets of the black boxes). In the case of the local approaches, the to be recognized instance x is marked with a black circle.

does not cover the entire feature value space, and the local real approach delivers too few instances in the vicinity of the instance x to be detected. In contrast, we can show that the local genetic neighborhood approach generates increased instances locally close to x. This increased instances should help to train a diff-classifier that can predict the local *difference-classes* accurately. The approach generates increased instances, especially in areas influenced by both black box models due to the fitness function $fitness^x_=$ and $fitness^x_{\neq}$ (see dark and dark gray marked areas in Figure 4.5c). Those more influenced areas have, therefore, a higher data density.

Finally, we show the data density plots of the generated local genetic neighborhood datasets (shown in Figure 4.3 and 4.4) in Figure 4.6a and 4.6b.



(a) Density of the local genetic neighborhood second dataset example

(b) Density of the local genetic neighborhood third dataset example

Figure 4.6: Data density of the local genetic neighborhood data approach examples (Figure 4.3b shows the datasets of the second dataset example (Seed 2) and Figure 4.4b shows the datasets of the third dataset example (Seed 3)). The to be recognized instances x are marked with a black circle.

4.5 Summary

In this chapter, we have explained in detail the approach of LORE to generate a genetic neighborhood close to an instance x. We also have presented how we adapt the approach of Guidotti et al. [GMR+18] to generate genetic instances to recognize differences between black box classifiers locally close to an instance x. We have shown different plots of created genetic neighborhood datasets with different located instances to be recognized. We have also compared the already presented datasets of the different data approaches to show how the genetic neighborhood approach can generate increased instances close to x. However, we have seen that the local genetic neighborhood approach generates unbalanced datasets and does not generate increased instances in the adjacent difference-class areas in all cases. The next chapter will show that the adapted genetic approach may be too simplistic, as evaluations will show.

CHAPTER 5

Evaluation of the Performance

In this chapter, the results are shown to find mainly answers for RQ 1 and its sub-questions. Therefore, we evaluate the performance of the various trained diff-classifiers and the already proposed different data approaches. Furthermore, we explain the simulation setting that enables the measurement of the performance in detail.

5.1 Experimental Setup

For measuring the performance of the diff-classifier, the *adult*, *bank marketing*, and *credit approval* UCI Machine Learning Repository datasets[1] are used [DG17]. The datasets used are available in the Gitlab-Repository[2]. The experiments are performed on Windows 20H2, 64 bit, 16 GB RAM, 2.60GHz Intel(R) Core(TM) i7-4510U and Python version 3.8.10 (64 bit). Furthermore, this section describes the preprocessing of the datasets and the manipulation of the datasets for the training of Black box B.

5.1.1 Presentation of the Used Datasets

The used *adult* multivariate classification dataset contains 32,561 instances. It includes 14 continuous and discrete features. The dataset contains the following continuous features: “age”, “fnlwgt”, “education-num”, “capital-gain”, “capital-loss”, and “hours-per-week”. The dataset contains the following discrete features: “workclass”, “edcuation”, “marital-status”, “occupation”, “relationship”, “race”, “sex”, and “native-country”.
The feature “workclass” contains the following different values: “Private”, “Self-emp-not-inc”, “Self-emp-inc”, “Federal-gov”, “Local-gov”, “State-gov”, “Without-pay”, and

[1]Sources of the datasets: *adult*: `https://archive.ics.uci.edu/ml/datasets/adult`, *bank marketing*: `https://archive.ics.uci.edu/ml/datasets/bank+marketing`, and *credit approval*: `https://archive.ics.uci.edu/ml/datasets/Credit+Approval`

[2]Available under: `https://gitlab.com/andsta/diro2c`

"Never-worked". The feature "education" contains the following different values: "Bachelors", "Some-college", "11th", "HS-grad", "Prof-school", "Assoc-acdm", "Assoc-voc", "9th", "7th-8th", "12th", "Masters", "1st-4th", "10th", "Doctorate", "5th-6th", and "Preschool".
The feature "marital-status" contains the following different values: "Married-civ-spouse", "Divorced", "Never-married", "Separated", "Widowed", "Married-spouse-absent", and "Married-AF-spouse".
The feature "occupation" contains the following different values: "Tech-support", "Craft-repair", "Other-service", "Sales", "Exec-managerial", "Prof-specialty", "Handlers-cleaners", "Machine-op-inspct", "Adm-clerical", "Farming-fishing", "Transport-moving", "Priv-house-serv", "Protective-serv", and "Armed-Forces".
The feature "relationship" contains the following different values: "Wife", "Own-child", "Husband", "Not-in-family", "Other-relative", and "Unmarried".
The feature "race" contains the following different values: "White", "Asian-Pac-Islander", "Amer-Indian-Eskimo", "Other", and "Black".
The feature "native-country" contains the following different values: "United-States", "Cambodia", "England", "Puerto-Rico", "Canada", "Germany", "Outlying-US(Guam-USVI-etc)", "India", "Japan", "Greece", "South", "China", "Cuba", "Iran", "Honduras", "Philippines", "Italy", "Poland", "Jamaica", "Vietnam", "Mexico", "Portugal", "Ireland", "France", "Dominican-Republic", "Laos", "Ecuador", "Taiwan", "Haiti", "Columbia", "Hungary", "Guatemala", "Nicaragua", "Scotland", "Thailand", "Yugoslavia", "El-Salvador", "Trinadad&Tobago", "Peru", "Hong", and "Holand-Netherlands".
Table 5.1 shows for each continuous feature μ, σ, the min-value, the max-value, the 25 percentile, the median, and the 75 percentile. Table 5.2 shows for each discrete feature the unique values, the top (most common) value, and the frequency of the most common value (how often occur the most common value). Depending on the income, the instance is assigned to the respective outcome class. The binary targets are "$<= 50K$" (0 class) and "$> 50K$" (1 class). The target class distribution is as follows: 24,720 instances are classified to class "$<= 50K$" and 7,841 instances are classified to class "$> 50K$". The dataset contains missing values which are marked with a "?" sign. There are a total of 4,262 missing values in the dataset. For our experiments, we are following the preprocess steps as explained below in detail. For the adult dataset, additionally, we are following the preprocess steps of Guidotti et al. in [GMR+18] and remove the following features: "fnlwgt" and "education-num". Guidotti et al. argue (comment) in their source code of LORE that the features are unnecessary.

The *bank marketing* dataset [MCR14] is also a multivariate classification dataset that contains 45,211 instances that are related to direct marketing campaigns of a Portuguese banking institute. It includes 16 continuous and discrete features. The dataset contains the following continuous features: "age", "balance", "duration" (description according to the source: "last contact duration, in seconds"), "campaign" (description according to the source: "number of contacts performed during this campaign and for this client"), "pdays" (description according to the source: "number of days that passed by after the client was

last contacted from a previous campaign"), and "previous" (description according to the source: "number of contacts performed before this campaign and for this client"). The dataset contains the following discrete features: "job", "marital", "education", "default" (description according to the source: "has credit in default"), "housing" (description according to the source: "has housing loan"), "loan", "contact", "month" (description according to the source: " last contact month of year"), "day_of_week" (description according to the source: "last contact day of the week"), and "poutcome" (description according to the source: "outcome of the previous marketing campaign").
The feature "job" contains the following different values: "admin.", "blue-collar", "entrepreneur", "housemaid", "management", "retired", "self-employed", "services", "student", "technician", "unemployed", and "unknown".
The feature "marital" contains the following different values: "divorced", "married", "single", "unknown", and "divorced". The feature "education" contains the following different values: "basic.4y", "basic.6y", "basic.9y", "high.school", "illiterate", "professional.course", "university.degree", and "unknown". The feature "default" contains the following different values: "no", "yes", and "unknown".
The feature "housing" contains the following different values: "no", "yes", and "unknown".
The feature "loan" contains the following different values: "no", "yes", and "unknown".
The feature "contact" contains the following different values: "cellular", and "telephone".
The feature "month" contains the following different values: "jan", "feb", "mar", "apr", "may", "jun", "jul", "aug", "sep", "oct", "nov", and "dec".
The feature "day_of_week" contains the following different values: "mon", "tue", "wed", "thu", and "fri".
The feature "poutcome" contains the following different values: "failure", "nonexistent", and "success".
Table 5.3 again shows for each continuous feature μ, σ, the min-value, the max-value, the 25 percentile, the median, and the 75 percentile. Talbe 5.4 shows for each discrete feature the unique values, the top (most common) value, and the frequency of the most common value (how often occur the most common value). The outcome class indicates if a client subscribed to a term deposit (bank product). Thus, there are two binary targets: "no" (0 class) and "yes" (1 class). The target class distribution is as follows: 39,922 instances are classified to class "no" and 5,289 instances are classified to class "yes". The dataset contains no missing values.

The *credit approval* dataset has the property that all attribute names are altered to not assignable designations to protect confidentiality. The authors of the dataset claim, it contains a good mix of continuous and discrete features with small and larger values. The 690 instances are classified into class "+" and "-", which means credit approved or credit not approved. It includes 15 continuous and discrete features. The continuous features are: "A2", "A3", "A8", "A11", "A14", and "A15". The discrete features are "A1", "A4", "A5", "A6", "A7", "A9", "A10", "A12", and "A13".
The feature "A1" contains the following different values: "b" and "a".
The feature "A4" contains the following different values: "u", "y", "l", and "t".

feature	count	μ	σ	min	25%	50%	75%	max
age	32561.00	38.58	13.64	17.00	28.00	37.00	48.00	90.00
capital-gain	32561.00	1077.65	7385.29	0.00	0.00	0.00	0.00	99999.00
capital-loss	32561.00	87.30	402.96	0.00	0.00	0.00	0.00	4356.00
hours-per-week	32561.00	40.44	12.35	1.00	40.00	40.00	45.00	99.00

Table 5.1: Continuous feature information of the adult dataset.

feature	count	unique	top	freq
workclass	32561	8	Private	24532
education	32561	16	HS-grad	10501
marital-status	32561	7	Married-civ-spouse	14976
occupation	32561	14	Prof-specialty	5983
realationship	32561	6	Husband	13193
race	32561	5	White	27816
sex	32561	2	Male	21790
native-country	32561	41	United-States	29753
income_class	32561	2	<=50K	24720

Table 5.2: Discrete feature information of the adult dataset.

The feature “A5” contains the following different values: “g”, “p”, and “gg”.
The feature “A6” contains the following different values: “c”, d”, “cc”, “i”, “j”, “k”, “m”, “r”, “q”, “w”, “x”, “e”, “aa”, and “ff”.
The feature “A7” contains the following different values: “v”, “h”, “bb”, “j”, “n”, “z”, “dd”, “ff”, and “o”.
The feature “A9” contains the following different values: “t” and “f”.
The feature “A10” contains the following different values: “t” and “f”.
The feature “A12” contains the following different values: “t” and “f”.
The feature “A13” contains the following different values: “g”, “p”, and “s”.
Table 5.5 again shows for each continuous feature μ, σ, the min-value, the max-value, the 25 percentile, the median, and the 75 percentile. Talbe 5.6 shows for each discrete feature the unique values, the top (most common) value, and the frequency of the most common value (how often occur the most common value). The outcome class indicates if a client subscribed to a term deposit (bank product). Thus, there are two binary targets: “+” (0 class) and “-” (1 class). The target class distribution is as follows: 307 instances are classified to class “+” and 383 instances are classified to class “-”. The dataset contains missing values which are marked with a “?” sign. There are a total of 67 missing values in the dataset.

feature	count	μ	σ	min	25%	50%	75%	max
age	45211.00	40.94	10.62	18.00	33.00	39.00	48.00	95.00
balance	45211.00	1362.27	3044.77	-8019.00	72.00	448.00	1428.00	102127.00
day	45211.00	15.81	8.32	1.00	8.00	16.00	21.00	31.00
duration	45211.00	258.16	257.53	0.00	103.00	180.00	319.00	4918.00
campaign	45211.00	2.76	3.10	1.00	1.00	2.00	3.00	63.00
pdays	45211.00	40.20	100.13	-1.00	-1.00	-1.00	-1.00	871.00
previous	45211.00	0.58	2.30	0.00	0.00	0.00	0.00	275.00

Table 5.3: Continuous feature information of the bank marketing dataset.

feature	count	unique	top	freq
job	45211	12	blue-collar	9732
marital	45211	3	married	27214
education	45211	4	secondary	23202
default	45211	2	no	44396
housing	45211	2	yes	25130
loan	45211	2	no	37967
contact	45211	3	cellular	29285
month	45211	12	may	13766
poutcome	45211	4	unknown	36959
y	45211	2	no	39922

Table 5.4: Discrete feature information of the bank marketing dataset.

5.1.2 Preprocessing of the Datasets

Each dataset is preprocessed by the following steps. The dataset, which is stored as a Comma-Separated Values (CSV) file, is read into a pandas dataframe. Besides, the pandas function to read a CSV dataset, marks the missing values to replace them in a later step. Afterward, we prepare the dataframe as follows: We detect for all features the data types. Finally, we assign all features to one of those data types: integer, double or string features. Then, we store for each feature the information whether the feature is continuous or discrete. This distinction is e.g. important when calculating the distance, since the distance for continuous and discrete features are calculated differently. In a next step, we prepare the dataframe and replace the marked, missing values for the continuous features by the mean and the discrete ones by calculating the mode. The discrete features are transformed by the sklearn class “LabelEncoder” that encodes the various possible discrete feature values by a integer value between 0 and the number of unique values of the particular discrete feature minus 1.

feature	count	μ	σ	min	25%	50%	75%	max
A2	690.00	31.51	11.86	13.75	22.67	28.46	37.71	80.25
A3	690.00	4.76	4.98	0.00	1.00	2.75	7.21	28.00
A8	690.00	2.22	3.35	0.00	0.17	1.00	2.63	28.50
A11	690.00	2.40	4.86	0.00	0.00	0.00	3.00	67.00
A14	690.00	183.56	172.19	0.00	80.00	160.00	272.00	2000.00
A15	690.00	1017.39	5210.10	0.00	0.00	5.00	395.50	100000.00

Table 5.5: Continuous feature information of the credit approval dataset.

feature	count	unique	top	freq
A1	690	2	b	480
A4	690	3	u	525
A5	690	3	g	525
A6	690	14	c	146
A7	690	9	v	408
A9	690	2	t	361
A10	690	2	f	395
A12	690	2	f	374
A13	690	3	g	625
A16	690	2	-	383

Table 5.6: Discrete feature information of the credit approval dataset.

5.1.3 Black Box Training and Manipulation

For each dataset, we train two decision tree-based classifiers using the Python scikit-learn library for imitating two binary black box classifiers A and B. We also use for both black box classifiers CART optimized algorithm[3] [HTF09, BFOS84]. We split the original and not modified dataset into a training and test dataset (with a ratio of 80 to 20 percent) with a configured random seed/state value. Black box A is trained with the original training dataset. After that, we manipulate the original dataset. We mainly have analyzed the resulting decision tree of black box A to detect for each dataset an influential feature. Besides, we also use the LIME method to double check the influence for each selected to be recognized instance. After that, we manipulate the values of the identified feature. The feature is manipulated by changing each value of the feature by a particular factor. We manipulate the datasets as follows:

[3]Sourcecode of the algorithm: `\https://github.com/scikit-learn/scikit-learn/blob/fdbaa58acbead5a254f2e6d597dc1ab3b947f4c6/sklearn/tree/tree.py#L584`

- *adult*: $x_{capital-gain} = x_{capital-gain} + 7000 \; \forall x \in X$. We adding the value 7,000 to Feature "capital-gain" for each instance x of X.
- *bank marketing*: $x_{pdays} = x_{pdays} + 4 \; \forall x \in X$. We adding the value 4 to Feature "pdays" for each instance x of X.
- *credit approval*: $x_{A10} = (x_{A10} + 1) \mod 2 \; \forall x \in X$. The discrete Feature "A10" contains 2 values "t" and "f", which are encoded as "0" and "1". To shift the value of the Feature "A10" by one for each x, x_{A10} plus 1 and then modulo the number of the possible values of x_{A10} is performed. Therefore, we shift the label encoded values 0 to 1 and the values 1 to 0.

In the case of the *adult* dataset, we want to explain in detail why we have chosen the described manipulation. We have selected some instances that are used for the simulations and investigated the most influential attribute with the aid of LIME and analysis of the resulting decision tree of the diff-classifier. In most cases, the feature "capital-gain" is the most influential attribute for the investigated instances. Subsequently, the dataset of black box B is manipulated by adding the value 7,000 to each instance of feature "capital-gain". After that, we have trained the decision tree-based diff-classifier by applying the simplistic genetic neighborhood approach to generate the diff-dataset. Then again, we split the manipulated dataset for black box B into a training and test dataset (with a ratio of 80 to 20 percent) with a configured random seed/state value. Black box B is trained by using the manipulated training dataset. Both test datasets of black box A and B are merged and passed to the DiRo2C method. This merged dataset is mainly used to calculate the value ranges for all features. In addition, the original, not manipulated entire dataset is passed to the DiRo2C method, which is used by the real data approaches. For each dataset, we randomly (with a defined random seed) pick 10 different instances of the test dataset of black box A.

Selection of the Test-Instances

We select the following indices of the *adult* test dataset of black box A: [2732, 2607, 1653, 3264, 4931, 4859, 5827, 1033, 4373, 5874]. The instances have the following feature values and outcome classes:

- Features of X_{Test}: ['age', 'workclass', 'education', 'marital-status', 'occupation', 'relationship', 'race', 'sex', 'capital-gain', 'capital-loss', 'hours-per-week', 'native-country']
- $X_{Test}[2732] : [25, 3, 2, 2, 4, 0, 3, 1, 0, 0, 60, 25]$, $Y_{Test}[2732] : [0]$
- $X_{Test}[2607] : [23, 5, 11, 4, 13, 3, 4, 1, 0, 0, 40, 38]$, $Y_{Test}[2607] : [0]$
- $X_{Test}[1653] : [42, 5, 7, 0, 2, 3, 0, 1, 0, 0, 35, 38]$, $Y_{Test}[1653] : [0]$

- $X_{Test}[3264] : [53, 3, 11, 2, 2, 0, 4, 1, 0, 0, 40, 38]$, $Y_{Test}[3264] : [1]$
- $X_{Test}[4931] : [35, 3, 11, 0, 13, 1, 4, 1, 0, 0, 60, 38]$, $Y_{Test}[4931] : [0]$
- $X_{Test}[4859] : [37, 3, 9, 2, 11, 0, 4, 1, 0, 1902, 40, 38]$, $Y_{Test}[4859] : [1]$
- $X_{Test}[5827] : [40, 4, 11, 2, 3, 0, 4, 1, 0, 0, 40, 38]$, $Y_{Test}[5827] : [0]$
- $X_{Test}[1033] : [38, 3, 15, 0, 3, 4, 4, 0, 0, 0, 40, 38]$, $Y_{Test}[1033] : [0]$
- $X_{Test}[4373] : [21, 3, 6, 4, 6, 1, 4, 1, 0, 0, 48, 25]$, $Y_{Test}[4373] : [0]$
- $X_{Test}[5874] : [60, 3, 7, 2, 11, 0, 4, 1, 0, 0, 30, 38]$, $Y_{Test}[5874] : [0]$

The discrete features and target of the *adult* dataset are encoded as follows:

- feature “workclass”: [0:8]: [‘Federal-gov’, ‘Local-gov’, ‘Never-worked’, ‘Private’, ‘Self-emp-inc’, ‘Self-emp-not-inc’, ‘State-gov’, ‘Without-pay’]
- feature “education”: [0:16]: [‘10th’, ‘11th’, ‘12th’, ‘1st-4th’, ‘5th-6th’, ‘7th-8th’, ‘9th’, ‘Assoc-acdm’, ‘Assoc-voc’, ‘Bachelors’, ‘Doctorate’, ‘HS-grad’, ‘Masters’, ‘Preschool’, ‘Prof-school’, ‘Some-college’]
- feature “marital-status”: [0:7]: [‘Divorced’, ‘Married-AF-spouse’, ‘Married-civ-spouse’, ‘Married-spouse-absent’, ‘Never-married’, ‘Separated’, ‘Widowed’]
- feature “occupation”: [0:14]: [‘Adm-clerical’, ‘Armed-Forces’, ‘Craft-repair’, ‘Exec-managerial’, ‘Farming-fishing’, ‘Handlers-cleaners’, ‘Machine-op-inspct’, ‘Other-service’, ‘Priv-house-serv’, ‘Prof-specialty’, ‘Protective-serv’, ‘Sales’, ‘Tech-support’, ‘Transport-moving’]
- feature “relationship”: [0:6]: [‘Husband’, ‘Not-in-family’, ‘Other-relative’, ‘Own-child’, ‘Unmarried’, ‘Wife’]
- feature “races”: [0:5]: [‘Amer-Indian-Eskimo’, ‘Asian-Pac-Islander’, ‘Black’, ‘Other’, ‘White’]
- feature “sex”: [0:2]: [‘Female’, ‘Male’]
- feature “native-country”: 00:‘Cambodia’, 01:‘Canada’, 02:‘China’, 03:‘Columbia’, 04:‘Cuba’, 05:‘Dominican-Republic’, 06:‘Ecuador’, 07:‘El-Salvador’, 08:‘England’, 09:‘France’, 10:‘Germany’, 11:‘Greece’, 12:‘Guatemala’, 13:‘Haiti’, 14:‘Holand-Netherlands’, 15:‘Honduras’, 16:‘Hong’, 17:‘Hungary’, 18:‘India’, 19:‘Iran’, 20:‘Ireland’, 24:‘Laos’, 25:‘Mexico’, 26:‘Nicaragua’, 27:‘Outlying-US(Guam-USVI-etc)’, 28:‘Peru’, 29:‘Philippines’, 30:‘Poland’, 31:‘Portugal’, 32:‘Puerto-Rico’, 33:‘Scotland’, 34:‘South’, 35:‘Taiwan’, 36:‘Thailand’
- target “income_class”: [0:2]: [‘<=50K’, ‘>50K’]

We select the following indices of the *bank marketing* test dataset of black box A: [2732, 3264, 4859, 7891, 4373, 5874, 6744, 3468, 705, 2599]. The instances have the following feature values and outcome classes:

- Features of X_{Test}: ['age', 'job', 'marital', 'education', 'default', 'balance', 'housing', 'loan', 'contact', 'day', 'month', 'duration', 'campaign', 'pdays', 'previous', 'poutcome']
- $X_{Test}[2732] : [25, 1, 2, 0, 0, 4599, 0, 0, 1, 13, 0, 120, 6, -1, 0, 3]$, $Y_{Test}[2732] : [0]$
- $X_{Test}[3264] : [29, 2, 1, 1, 0, 291, 1, 0, 2, 2, 6, 205, 3, -1, 0, 3]$, $Y_{Test}[3264] : [0]$
- $X_{Test}[4859] : [45, 1, 1, 0, 0, 1297, 1, 0, 2, 6, 8, 233, 2, -1, 0, 3]$, $Y_{Test}[4859] : [0]$
- $X_{Test}[7891] : [50, 4, 1, 2, 0, 15442, 0, 0, 2, 5, 6, 91, 1, -1, 0, 3]$, $Y_{Test}[7891] : [0]$
- $X_{Test}[4373] : [40, 1, 2, 1, 0, 677, 1, 0, 017, 9, 619, 1, 171, 5, 1]$, $Y_{Test}[4373] : [0]$
- $X_{Test}[5874] : [51, 7, 1, 1, 0, 272, 0, 1, 0, 8, 5217, 1, -1, 0, 3]$, $Y_{Test}[5874] : [0]$
- $X_{Test}[6744] : [58, 4, 1, 2, 0, 16264, 0, 0, 1, 17, 9, 215, 3, -1, 0, 3]$, $Y_{Test}[3468] : [0]$
- $X_{Test}[3468] : [47, 0, 1, 1, 0, 310, 1, 1, 1, 14, 1, 128, 6, -1, 0, 3]$, $Y_{Test}[1033] : [0]$
- $X_{Test}[705] : [36, 4, 2, 2, 0, 188, 0, 0, 0, 29, 4, 55, 2, -1, 0, 3]$, $Y_{Test}[705] : [0]$
- $X_{Test}[2599] : [50, 9, 1, 1, 0, 4, 0, 0, 0, 10, 5, 200, 5, -1, 0, 3]$, $Y_{Test}[2599] : [0]$

The discrete features and target of the *bank marketing* dataset are encoded as follows:

- feature "job": [0:12]: ['admin.', 'blue-collar', 'entrepreneur', 'housemaid', 'management', 'retired', 'self-employed', 'services', 'student', 'technician', 'unemployed', 'unknown']
- feature "marital": [0:3]: ['divorced', 'married', 'single']
- feature "education": [0:4]: ['primary', 'secondary', 'tertiary', 'unknown']
- feature "default": [0:2]: ['no', 'yes']
- feature "housing": [0:2]: ['no', 'yes']
- feature "loan": [0:2]: ['no', 'yes']
- feature "contact": [0:3]: ['cellular', 'telephone', 'unknown']
- feature "month": [0:12]: ['apr', 'aug', 'dec', 'feb', 'jan', 'jul', 'jun', 'mar', 'may', 'nov', 'oct', 'sep']

- feature “outcome”: [0:4]: [‘failure’, ‘other’, ‘success’, ‘unknown’]
- target “y”: [0:2]: [‘no’, ‘yes’]

We select the following indices of the *credit approval* test dataset of black box A: [47, 117, 67, 103, 9, 21, 36, 87, 70, 88]. The instances have the following feature values and outcome classes:

- Features of X_{Test}: [‘A1’, ‘A2’, ‘A3’, ‘A4’, ‘A5’, ‘A6’, ‘A7’, ‘A8’, ‘A9’, ‘A10’, ‘A11’, ‘A12’, ‘A13’, ‘A14’, ‘A15’]
- $X_{Test}[47] : [1, 40.58, 1.5, 1, 0, 6, 0, 0, 0, 0, 0, 0, 2, 300, 0]$, $Y_{Test}[47] : [1]$
- $X_{Test}[117] : [1, 23.5, 2.75, 1, 0, 5, 2, 4.5, 0, 0, 0, 0, 0, 160, 25]$, $Y_{Test}[117] : [1]$
- $X_{Test}[67] : [1, 22.83, 3, 1, 0, 9, 7, 1.29, 1, 1, 1, 0, 0, 260, 800]$, $Y_{Test}[67] : [0]$
- $X_{Test}[103] : [1, 43.17, 2.25, 1, 0, 6, 0, 0.75, 1, 0, 0, 0, 0, 560, 0]$, $Y_{Test}[103] : [1]$
- $X_{Test}[9] : [0, 22.67, 0.79, 1, 0, 6, 7, 085, 0, 0, 0, 0, 0, 144, 0]$, $Y_{Test}[9] : [1]$
- $X_{Test}[21] : [1, 178, 3.29, 1, 0, 6, 7, 0.335, 0, 0, 0, 1, 0, 140, 2]$, $Y_{Test}[21] : [1]$
- $X_{Test}[36] : [1, 19.5, 9.585, 1, 0, 0, 7, 0.79, 0, 0, 0, 0, 0, 80, 350]$, $Y_{Test}[36] : [1]$
- $X_{Test}[87] : [1, 348, 6.5, 1, 0, 0, 7, 0.125, 1, 0, 0, 1, 0, 443, 0]$, $Y_{Test}[87] : [1]$
- $X_{Test}[70] : [1, 36.67, 2, 1, 0, 6, 7, 0.25, 0, 0, 0, 1, 0, 221, 0]$, $Y_{Test}[70] : [1]$
- $X_{Test}[88] : [1, 26.67, 4.25, 1, 0, 2, 7, 4.29, 1, 1, 1, 1, 0, 120, 0]$, $Y_{Test}[88] : [0]$

The discrete features and target of the *credit approval* dataset are encoded as follows:

- feature “A1”: [0:2]: [‘a’, ‘b’]
- feature “A4”: [0:3]: [‘l’, ‘u’, ‘y’]
- feature “A5”: [0:3]: [‘g’, ‘gg’, ‘p’]
- feature “A6”: [0:14]: [‘aa’, ‘c’, ‘cc’, ‘d’, ‘e’, ‘ff’, ‘i’, ‘j’, ‘k’, ‘m’, ‘q’, ‘r’, ‘w’, ‘x’]
- feature “A9”: [0:9]: [‘bb’, ‘dd’, ‘ff’, ‘h’, ‘j’, ‘n’, ‘o’, ‘v’, ‘z’]
- feature “A10”: [0:2]: [‘f’, ‘t’]
- feature “A11”: [0:2]: [‘f’, ‘t’]
- feature “A13”: [0:3]: [‘g’, ‘p’, ‘s’]

- target “A16”: [0:2]: [‘+’, ‘-’]

Therefore, we have to test three datasets, and each dataset instantiates the four various data approaches. These are in turn applied to the two binary and multiclass diff-classifier methods. For each test run, the diff-dataset is created. Next, we present in detail the performance evaluation of the trained diff-classifiers.

5.2 Performance Evaluation of Classifiers

For measuring the performance of the diff-classifier, we use the following setting: Random seed/state values are applied to prevent randomness as much as possible and to enable reproducibility and scientific rigor. DiRo2C trains depending on the data approach the diff-classifier. We evaluate the trained diff-classifier by applying a stratified k-fold cross-validator (where $k = 10$) to split the diff-dataset and calculate the specific performance metrics. Finally, we compute the mean and standard deviation over all 10 test runs. Since, stratified means that the instances are divided up equally that each fold preserve approximately the same percentage of each outcome class as the entire dataset, we check for every testrun the target list Y_{test} of the diff-dataset. If a outcome class is assigned to fewer than 10 instances, k is reduced to this number. Both outcome classes must appear in the list at least twice that the test-run is counted. Otherwise the test-run is unrated.

The following metrics and measures are mainly considered: *Accuracy*, F_1-*score*, and $PearsonCC$. The measured metrics are defined as follows [CJ20, HM15, Rij79, Pow08]. The *Accuracy* metric (see Equation 5.1) calculation of sklearn is defined as:

$$ACC(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i) \tag{5.1}$$

Whereby y denotes the real (actual) values and $\hat{y}$ the predicted value of the classifier. Thus, it calculates the portion of correct predicted results over $n_{samples}$ ($n_{instances}$).
As already described in Section 3.2, the difference detections of the binary diff-classifier are dedicated to the positive 1 outcome class. That is particularly important in the case of the binary diff-classifier method, since F_1-*score* for binary classification (see Equation 5.2) is defined as the weighted harmonic mean of the precision and recall metric:

$$F_1\text{-}score = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} = \frac{2 \cdot Prec \cdot Rec}{Prec + Rec} \tag{5.2}$$

Whereby TP denotes the true positive, FP the false positive, and FN the false negative predicted results. $Prec$ is the precision, and Rec the recall value. In the case of the multiclass diff-classifier, we calculate the macro F_1-*score*, which calculates the value for each outcome class and determines their unweighted mean. Since the F_1-*score* is highly dependent on the TP, we can evaluate how accurately the diff-classifier can predict the

differences. Finally, to confirm the results we measure the *PearsonCC* metric, which in sklearn for the binary case (see Equation 5.3) is defined as follows:

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{5.3}$$

The *PearsonCC* is a balanced measure and returns reliable results even if the datasets are unbalanced. A correlation coefficient value of 1 means a perfect predicted result. The *PearsonCC* for the multiclass case (see Equation 5.4) is defined by sklearn for K classes as follows:

$$MCC = \frac{c \cdot s - \sum_k^K p_k \cdot t_k}{\sqrt{(s^2 - \sum_k^K \cdot p_k^2) \cdot (s^2 - \sum_k^K t_k^2)}} \tag{5.4}$$

Where $t_k = \sum_i^K C_{ik}$ and determines how often the class k is actually assigned, $p_k = \sum_i^K C_{ki}$ which is the number how often the class k is predicted, c denotes how often instances are correctly predicted: $c = \sum_k^K C_{kk}$, and $s = \sum_i^K \sum_j^K C_{ij}$ denotes the total number of instances.

5.3 Results and Findings

Before we present our results, we want to list all unrated test-runs per dataset. In the following test-runs, the approaches generate a diff-dataset that contains too few instances for a particular outcome class. As a result, those test-runs can not be included in the performance measurement. The following test-runs with the respective indices of the to be recognized instances are unrated:

- *adult* dataset:
  - binary local genetic neighborhood: 1653, 4931, 1033
  - binary local real data: 2732, 2607, 3264, 4931, 4859, 5827, 1033, 4373, 5874
  - multiclass local genetic neighborhood: 1653, 4931
  - multiclass local real data: 2732, 2607, 1653, 3264, 4931, 5827, 1033, 4373, 5874
- *bank marketing* dataset:
  - binary local genetic neighborhood: 2732, 7891, 4373, 6744, 2599
  - binary local real data: 2732, 3264, 4859, 7891, 5874, 6744, 3468, 705, 2599
  - multiclass local genetic neighborhood: 2732, 4373, 3468, 705, 2599
  - multiclass local real data: 2732, 3264, 4859, 7891, 4373, 5874, 6744, 3468, 2599

- *credit approval* dataset:
  - binary local genetic neighborhood: 67

Also, the genetic neighborhood approach generates in some cases too few instances for a particular outcome class. In the case of the binary setting, mostly too few instances for the "diff" difference-class are generated. Also, in the case of the multiclass setting, too few instances for the "10" or/and "01" difference-classes are generated. But it is not the only approach that generates (provides) a dataset with a poor class distribution. We can observe that only local data approaches are involved in all unrated test-runs. Therefore, the datasets of the local data approaches contain not in every case enough instances which are predicted differently by the black boxes. The fact that some test-runs are not included in the calculation of the performance metrics must be considered by interpreting the results.

Now, we present in Table 5.7 the measured metrics.

According to RQ 1 and our hypothesis, the Accuracy of the binary diff-classifiers is, in all cases, as expected, higher than for the multiclass diff-classifiers. The F_1-*score* does not confirm this. In some cases, the F_1-*score* of the multiclass classifier is better here. It is noticeable that the genetic neighborhood approach performs, according to the F_1-*score*, relatively poorly for the *adult* and *bank marketing* datasets compared to the *credit approval* dataset. We can also see that the multiclass local genetic neighborhood approach, at least in the case of the *adult* and *bank marketing* dataset, performs better than the binary local genetic neighborhood approach in terms of the Pearson CC-score. Therefore, in that case, the Pearson CC-score cannot confirm that the binary approach recognizes the differences more precisely. But, according to RQ 1.1, we can confirm that a classifier trained on synthetic data can outperform a classifier trained using real data, especially in the case of the local genetic neighborhood approach. According to RQ 1.2, we can also confirm that a local approach, especially in the case of the local genetic neighborhood approach, performs overall better to predict the difference-classes compared to the global approaches. It should also be mentioned that the local real datasets lead, in most cases, to unrated test-runs. Therefore, the comparability of the metrics for this approach, especially for the *adult* and *bank marketing* datasets, should be interpreted with caution. However, this shows us this approach can generate, in most cases, too few instances that are predicted differently by black box A and B.
Furthermore, one main finding is, that the local genetic neighborhood approach does not always generate instances that are predicted differently by black box A and B. A closer analysis of the *adult* dataset reveals some reasons why the performance for the *adult* and *bank marketing* dataset is worse than expected. Figure 5.1 shows the resulting decision tree of the diff-classifier of one particular case. Interestingly, the predictor classifies 1,977 instances of the overall 2,191 instances as "no diff" at the first branch. Overall, the outcome class distribution is as follows: 2,159 instances are identified as

Dataset	DCM	DA	*ACC*	F_1-*score*	*PCC*
adult	binary	gn	.984 ± .006	.703 ± .055	.695 ± .073
		lr	.999 ± .000	.833 ± .000	.853 ± .000
		gs	.977 ± .000	.583 ± .006	.574 ± .005
		gr	.981 ± .000	.551 ± .007	.540 ± .007
	multiclass	gn	.906 ± .045	.667 ± .039	.814 ± .089
		lr	.779 ± .000	.389 ± .000	.557 ± .000
		gs	.825 ± .001	.625 ± .005	.547 ± .002
		gr	.858 ± .001	.633 ± .006	.610 ± .001
bank marketing	binary	gn	.996 ± .003	.711 ± .357	.717 ± .361
		lr	.982 ± .000	.240 ± .000	.196 ± .000
		gs	.990 ± .000	.147 ± .010	.139 ± .009
		gr	.991 ± .000	.144 ± .008	.137 ± .011
	multiclass	gn	.931 ± .015	.759 ± .126	.814 ± .089
		lr	.853 ± .000	.317 ± .000	.254 ± .000
		gs	.874 ± .001	.453 ± .006	.432 ± .003
		gr	.878 ± .001	.441 ± .003	.437 ± .003
credit approval	binary	gn	.983 ± .009	.911 ± .058	.900 ± .044
		lr	.873 ± .005	.575 ± .022	.499 ± .025
		gs	.880 ± .001	.521 ± .007	.459 ± .007
		gr	.849 ± .002	.519 ± .020	.441 ± .015
	multiclass	gn	.979 ± .012	.897 ± .050	.966 ± .015
		lr	.769 ± .004	.629 ± .009	.643 ± .006
		gs	.841 ± .002	.685 ± .007	.737 ± .005
		gr	.764 ± .005	.629 ± .008	.634 ± .010

Table 5.7: Diff-classifier performance results. The following abbreviations are used: DCM (Diff-Classifier Method), DA (Data Approach), *ACC* (Accuracy), *PCC* (Pearson CC) and the abbrevations for the data approaches: gn (local genetic neighborhood), lr (local real data), gs (global synthetic data), and gr (global real data). The cells with a red background mark the poor performance of the genetic neighborhood approach.

"no diff" and only 32 instances as "diff". The F_1-*score* of the diff-classifier trained for a particular test-instance x is 0.655 (the average of 10 test runs is: 0.703), which is also significantly less compared to other test runs applied to other datasets. The problem is that the original local genetic neighborhood approach is applied for black box A and B independently. Therefore the method does not find enough differences since there is no explicit lookup whether black box A and B are predicting different results. The to be examined instance x has for the feature "capital-gain" the value 0. However, the local

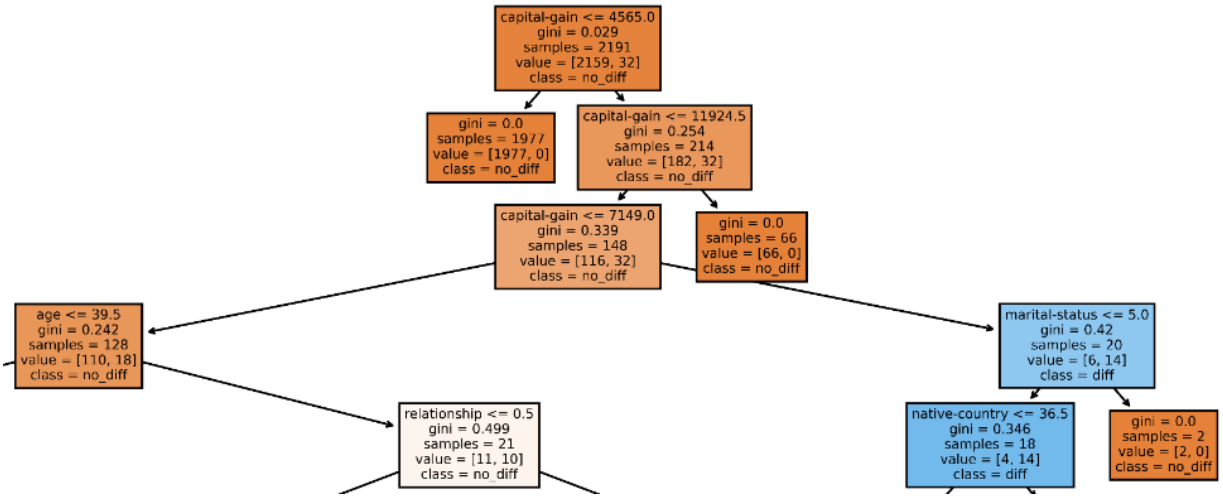


Figure 5.1: Decision tree of the diff-classifier for identifying the weak spot of the genetic neighborhood approach for DiRo2C

simplistic genetic neighborhood approach generates too few instances for black box A and B with a value greater than 4,565 for the attribute "capital-gain". Thus, it generates too few instances for which black box A and B predict different results. That means this approach provides the dataset Z_{Diff} for one black box (independent of the other) with 50 percent instances where the black box predicts the target $y = 0$ and 50 percent instances where $y = 1$. As a result, too few instances are generated where the black boxes return different results: $y_A <> y_B$.

5.4 Summary

In general, the already presented data approaches do not guarantee datasets with a balanced difference-class distribution. We also can confirm it in our analyzes. Furthermore, another problem is that the genetic neighborhood approach generates the instances for black boxes A and B independently. In general, we find out that the local data approaches provide, in many cases, datasets that contain too few instances that are predicted by black box A and B differently. Thus, it is not guaranteed that the approach generates instances where black box A and B predict different outcomes. So, we will introduce a modified genetic neighborhood approach in the next chapter and present an approach to the described problem of unbalanced diff-datasets, especially in the case of the binary difference classification problem and the independent instance generation.

CHAPTER 6

Modified Genetic Neighborhood and Performance Evaluation

This chapter describes an improved genetic neighborhood approach to generate a balanced diff-dataset Z_{Diff} to learn the differences more accurately. As already described in Chapter 4, LORE creates balanced instances $z \in Z_{=} \cup Z_{\neq}$ by using a particular genetic algorithm approach and maximizing the original fitness function. But the problem-solving method is only aimed at one black box to create a neighborhood to explain the predictions of the black box for a particular instance x.

6.1 Modification of the Genetic Neighborhood Approach

Next, we show how we modified the original fitness function that the instances are generated for black box A and B dependent on each other. Therefore, we modify the original fitness functions of LORE (cf. Equation 4.1 and 4.2) . In Equation 6.1 and 6.2 we present the modified fitness functions:

$$modifiedFitness^{x}_{=}(z) = I_{b_A(z)=b_B(z)} + (1 - d(x, z)) - I_{x=z} \tag{6.1}$$

$$modifiedFitness^{x}_{\neq}(z) = I_{b_A(z)\neq b_B(z)} + (1 - d(x, z)) - I_{x=z} \tag{6.2}$$

where d is again a distance function: $d : \mathcal{X}^{(m)} \rightarrow [0, 1]$ and $I = 1$, if the condition is *true*, else $I = 0$. In both cases, the fitness functions search for instances where the feature characteristics are close to x: $(1 - d(x, z))$ but not equal to x (see term $I_{x=z}$). The fitness function defined in Equation 6.1 looks especially for instances where the black box A b_A returns for a potential instance z_0 the same outcome as the black box B b_B. In

contrast, the function defined in Equation 6.2 guarantees the generation of instances for which b_A predicts a different outcome than b_B. Thus, the function $modifiedFitness^{x}_{=}(z_0)$ for an instance z_0 where $b_A(z_0) \neq b_B(z_0)$ and $x \neq z_0$ leads to a result smaller than 1. Instead, $b_A(z_0) = b_B(z_0)$ leads to a result greater or equal than 1. For the to be recognized instance x itself $modifiedFitness^{x}_{=}(x)$ returns either 0 (if black box A and B predict a different outcome) or 1 (if black box A and B predict the same outcome). Therefore, the function guarantees again the generation of instances that are different from x and are as close as possible to x. The crucial difference to the original $fitness$ function of LORE is, that the $modifiedFitness$ function is depending on black box A and B. Thus, in the case of the binary difference classification, it is also guaranteed that the genetic generation algorithm creates balanced sets with an approximate proportion of 50 percent instances where the black boxes predict different outcomes and 50 percent instances where the black boxes return the same outcomes. The generation of balanced sets is impossible if the black boxes are identical because no instances would be predicted differently. In the case of the multiclass difference classification, balanced sets in relation to the various multiclass difference-classes would also be desirable. But first, it is not guaranteed because a particular multiclass difference-class does not always have to occur. Second, the modified genetic neighborhood approach does not guarantee the balanced generation of all possible multiclass difference-classes generally, although all multiclass difference-classes would occur. The reason for this is that the fitness functions only ensure that roughly the same number of instances are generated that are either predicted equally by the black boxes or are predicted differently by the black boxes.

We also change the initial adapted Algorithm 4.3, which creates a neighborhood Z_A for black box A and one Z_B for black box B independently, as presented in Algorithm 6.1. The modified approach (as the original approach of LORE) aims to find and create a set of Z, with feature values and characteristics similar to the to be recognized instance x to approximate the local behavior of the black boxes. We use the "ModifiedGeneticNeigh" function proposed in Algorithm 6.2 to generate the set $Z_=$ (see line number 2) that includes instances where the black boxes predict the same results and the set $Z_{\neq}$ (see line number 3) that includes instances where the black boxes returns a different outcome. The generation of the neighborhood is now dependent on both black boxes A and B. Whereby, the instances $z \in Z_=$ have to meet the condition $b_A(z) = b_B(z)$ and the instances $z \in Z_{\neq}$ the following condition $b_A(z) \neq b_B(z)$. After that, we merge the created instances $Z_=$ and $Z_{\neq}$: $Z = Z_= \cup Z_{\neq}$ (see line number 4). Z and the instances it contains are finally checked for uniqueness and the redundant instances are removed. Thus, the cleaned Z no longer guarantees a balanced dataset in all cases. Then, we use the set Z to determine the target Y (cf. Section 3.2) for the diff-dataset (see line number 5). For building the diff-dataset the modified approach calls again the already presented $BuildDiffDataset$ function (cf. Algorithm 3.1). After that, it trains the diff-classifier based on the diff-dataset Z_{Diff} (see line number 6). Finally, it returns the dataset Z_{Diff} and the decision tree classifier to detect the differences between black box A and B (see line number 7).

Algorithm 6.1: $DiRo2C\_MGN(x, b_A, b_B, N)$, Modified from: [GMR+18]

Input: x - instance to explain, b_A - black box A, b_B - black box B, N - # of neighbors

Output: dc - diff-classifier, Z_{Diff} - generated neighborhood depending on the black boxes and determined target Y

1 $G \leftarrow 10$; $pc \leftarrow 0.5$; $pm \leftarrow 0.2$; `// init parameters`
`// generate neighbors depending on black box A and B`
2 $Z_{=} \leftarrow ModifiedGeneticNeigh(x, modifiedFitness^{x}_{=}, b_A, b_B, N/2, G, pc, pm)$;
3 $Z_{\neq} \leftarrow ModifiedGeneticNeigh(x, modifiedFitness^{x}_{\neq}, b_A, b_B, N/2, G, pc, pm)$;
4 $Z \leftarrow Z_{=} \cup Z_{\neq}$; `// merge neighborhoods`
5 $Z_{Diff} \leftarrow BuildDiffDataset(b_A, b_B, Z)$; `// determine target Y`
6 $dc \leftarrow TrainDecisionTreeClassifier(Z_{Diff})$; `// train diff-classifier`
7 **return** dc, Z_{Diff};

Algorithm 6.2: $ModifiedGeneticNeigh(x, modifiedFitness, b_A, b_B, N, G, pc, pm)$, Adapted from: [GMR+18]

Input: x - instance to explain, b_A - black box A, b_B - black box B, $modifiedFitness$ - modified fitness function, N - population size, G - # of generations, pc - crossover probability, pm - mutation probability

Output: Z - neighbors of x

1 $P_0 \leftarrow \{x|\forall 1...N\}$; $i \leftarrow 0$; `// population init`
2 $evaluate(P_0, modifiedFitness, b_A, b_B)$; `// evaluate population`
3 **while** $i < G$ **do**
4 $P_{i+1} \leftarrow select(P_i)$; `// select sub-population`
5 $P'_{i+1} \leftarrow crossover(P_{i+1}, pc)$; `// mix records`
6 $P''_{i+1} \leftarrow mutate(P'_{i+1}, pm)$; `// perform mutations`
7 $evaluate(P''_{i+1}, modifiedFitness, b_A, b_B)$; `// evaluate population`
8 P_{i+1}=P''_{i+1}; $i \leftarrow i + 1$; `// update population`
9 **end**
10 $Z \leftarrow P_i$;
11 **return** Z;

As already mentioned, we implement and modify the genetic algorithm of LORE to generate a balanced genetic neighborhood (see Algorithm 6.2). The main difference to the original approach is that the $modifiedFitness$ function and the two black boxes are now passed to the *evaluate* function (see line number 2 and 7). Thus a balanced dataset Z can be created (see line number 10). Otherwise, we continue to use the biologically inspired original implemented *crossover* (see line number 5) and *mutation* (see line number 6) operations of LORE.

The following section shows how differently the modified approach generates the synthetic

diff-dataset Z_{Diff}.

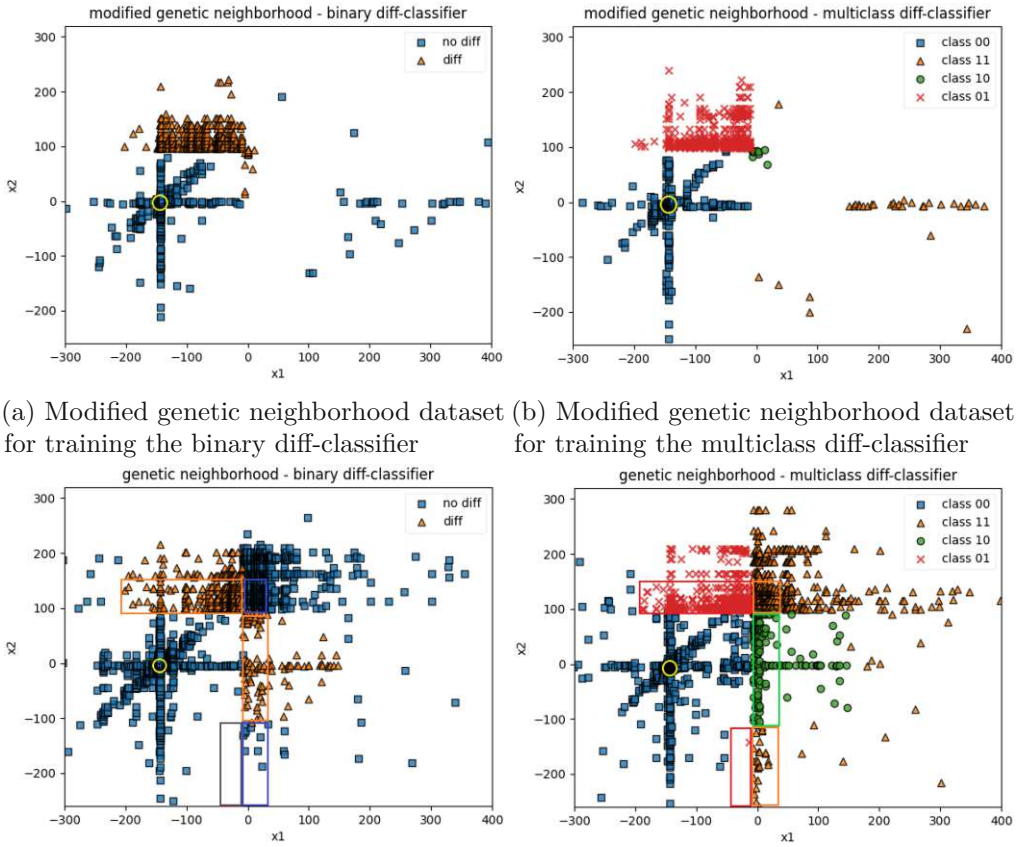
6.2 Effects of the Modified Genetic Neighborhood Approach

Now, we present the effects of the modified genetic neighborhood approach. We first use our running example, the two-dimensional datasets presented in Chapter 3, to show the effects. Then, we present two different parametrized Gaussian quantiles datasets for black box A and B to show the effects from a different perspective. We also show the diff-datasets generated by the modified genetic neighborhood approach and the diff-datasets from the initial genetic neighborhood approach for comparison. After each presented generated datasets, we show the data density plots of the two genetic neighborhood approaches. In this section, we again use for the data density plots bins to group the instances on the axes. For the x_1-axis, we use 80 bins, and for the x_2-axis, we use 60 bins. The data density is scaled from 0 to 100 and represents the counts of instances per bin. The white areas show bins where no instances exist and to be recognized instance x is marked with a black circle.

6.2.1 Running Example Dataset

At first, we present the effects of the modified approach by again using our running example.

Seed 1: Figures 6.1a and 6.1b illustrate the datasets generated by the modified genetic neighborhood approach to recognize local differences close to an instance x between two binary black boxes. For this example, we use the already presented running example classification datasets of black box A and black box B (Figure 3.5 shows the datasets of the black box classifiers). The datasets for the binary and multiclass classifier are generated independently of each other, which means the instances (data points) of the datasets shown in the plots are not identical. For comparison, we also show the generated datasets of the initial genetic neighborhood approach in Figures 6.1c and 6.1d. The now balanced dataset for the binary diff-classifier (see Figure 6.1a) contains 818 instances. 413 instances are assigned to the class “0” (“no diff”), and 405 instances are assigned to the class “1” (“diff”). The dataset for the multiclass diff-classifier (see Figure 6.1b) contains 864 instances, whereby 384 instances are classified into class “00”, 35 instances are classified into class “11”, 10 instances are classified into class “10”, and 435 instances are classified into class “01”. Whereby, the combined instances of the class “00” and “11” compared to the combined instances of class “10” and “01” are balanced. Each dataset is generated based on the instance x highlighted in yellow, and both instances are located at: $x_1 = -143.91$ and $x_2 = -4.53$. Based on x, synthetic instances for both black boxes are now generated dependent on each other. By applying the modified genetic algorithm to generate a neighborhood for black box A and B, the aim again is to generate increased



(a) Modified genetic neighborhood dataset for training the binary diff-classifier

(b) Modified genetic neighborhood dataset for training the multiclass diff-classifier

(c) Genetic neighborhood dataset for training the binary diff-classifier

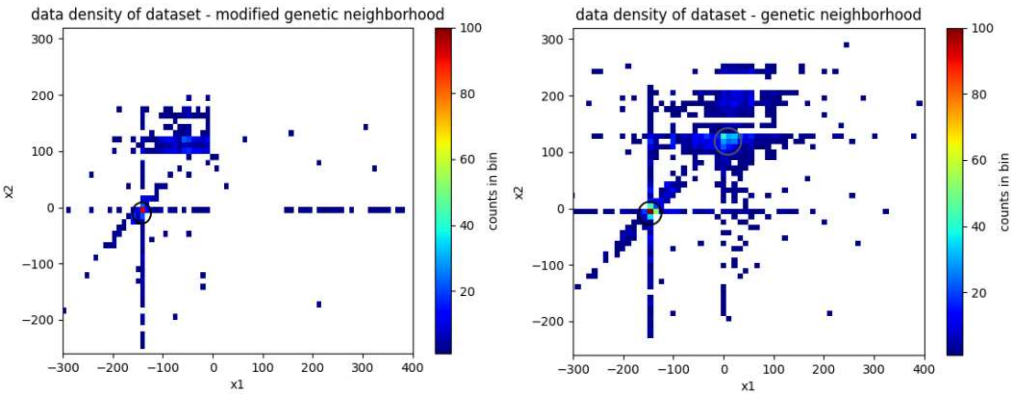
(d) Genetic neighborhood dataset for training the multiclass diff-classifier

Figure 6.1: Running example: Datasets generated by the genetic neighborhood approaches with the initial to be recognized located instance (Seed 1) for training a binary and a multiclass diff-classifier to recognize local differences close to instance x between black box A and B (Figure 3.5 shows the datasets of the black boxes). Depending on the yellow-marked instance, the local modified genetic neighborhood approach (see Algorithm 6.1) of DiRo2C generates the shown dataset.

instances close to the yellow marked instance x to recognize the local boundaries where the instances are classified into different *difference-classes*.

Figure 6.1a and 6.1b, shows that the modified genetic neighborhood approach compared to the genetic neighborhood approach generates increased instances as close as possible to the to be recognized instance x. The approach generates increased instances close to x where the black boxes predict the same outcome and increased instances close to x where the black boxes predict a different outcome. That also means that in some areas, few or no instances are generated. Figure 6.1a shows the dataset for the binary difference classification. It shows that in the upper left area too few instances are generated to

differentiate between class “no diff” and “diff”. That is also recognizable in the middle area of the plot, where generated instances would be classified into class “diff”. In all of those more distant areas, few or no instances are generated. Figure 6.1a shows the dataset for the multiclass difference classification. Again increased instances are generated to recognize the boundary between outcome class “00” and “11”. For all other areas, too few or no instances are generated. That leads to an unbalanced dataset with few generated instances which are assigned to the opposite outcome classes “10” and “01”. For comparing the genetic neighborhood approaches we show again in Figures 6.1c and 6.1d the datasets generated by the genetic neighborhood approach.
Figure 6.2 shows the data density plots of the generated datasets of both approaches. Both approaches generate increased instances which are predicted by black box A and B with the same result and which are locally around instance x. In contrast to the initial genetic neighborhood approach, the modified approach generates increased instances which are predicted by black box A and B differently and locally close to instance x.



(a) Density of the local modified genetic neighborhood (Seed 1)

(b) Density of the local genetic neighborhood (Seed 1)

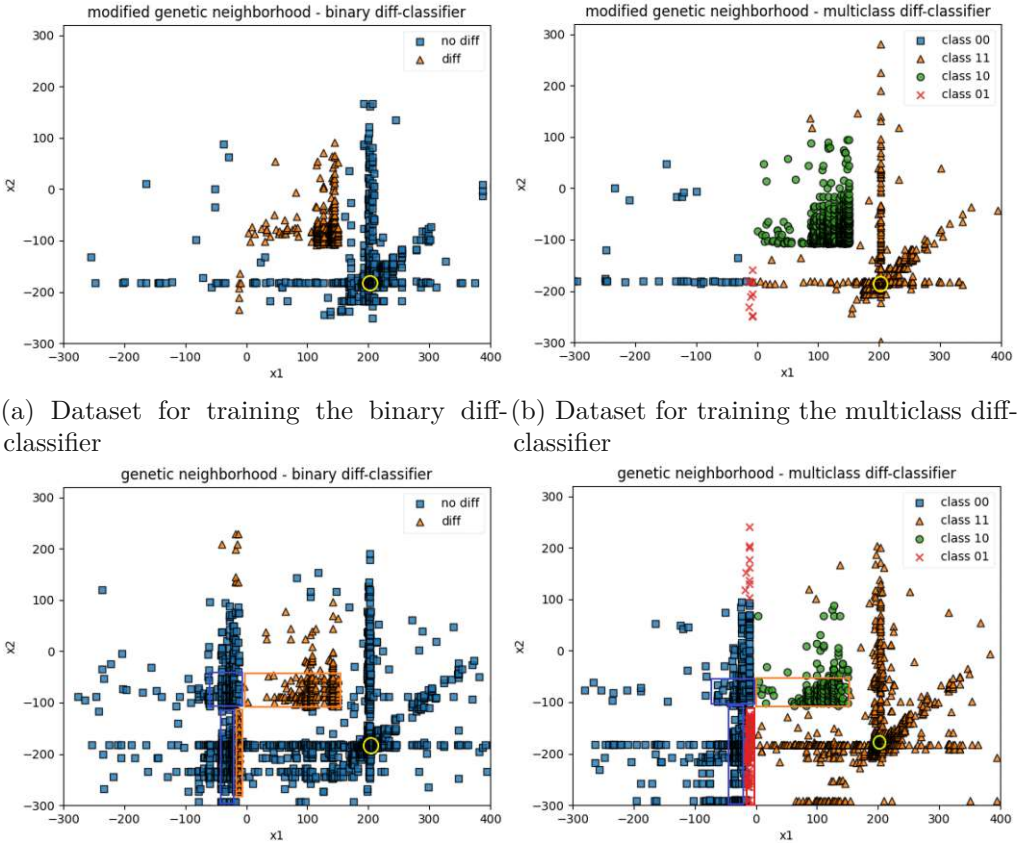
Figure 6.2: Data density of both local genetic neighborhood approaches. Figure 6.1 shows the generated datasets of the modified approach for Seed 1. The to be recognized instances x are marked with a black circle.

Seed 2: Now, we show in our second example, a dataset based on our running example but generated with a different seed, where we choose the to be recognized instance x differently. The datasets, shown in Figures 6.3a and 6.3b, are based on the yellow marked instances x and both are located at: $x_1 = 203.73$ and $x_2 = -182.83$. The datasets for the binary and multiclass classifier are generated independently of each other, which means the instances (data points) of the datasets shown in the plots are not identical. For comparison, we show the generated datasets of the initial genetic neighborhood approach in Figures 6.3c and 6.3d. The balanced dataset for the binary diff-classifier contains 1,046 instances. 542 instances are assigned to the class “0” (“no diff”), and 504 instances are assigned to the class “1” (“diff”). The dataset for the multiclass diff-classifier contains

1,145 instances, whereby 36 instances are classified into class “00”, 415 instances are classified into class “11”, 685 instances are classified into class “10”, and only 9 instances are classified into class “01”. Figure 6.3a shows the binary diff-dataset. We can observe that the approach generates, depending on instance x, very few instances classified into class “no diff” near the closer boundary to the class “diff”. That can also be observed in the multiclass example shown in Figure 6.3b. In that case, very few instances with the class “11” are generated near the decision boundary to the class “11”. In summary, increased instances are generated around the boundary to the adjacent classes in the middle area of the plot (as in the binary as well as in the multiclass setting). That should help to recognize the differences in that area. Again, in all of the more distant areas compared to instance x, few or no instances are generated. For comparing the genetic neighborhood approaches we show again in Figures 6.3c and 6.3d the datasets generated by the genetic neighborhood approach.

Figure 6.4 shows the data density plots of the generated datasets of both approaches. This example with a different seed shows that both approaches generate increased instances which are predicted by black box A and B with the same result and which are locally around instance x. But, as already mentioned, we can observe that both approaches generate, depending on the instance x, very few instances classified into class “no diff” near the closer boundary to the class “diff”. In contrast to the initial genetic neighborhood approach, the modified approach generates increased instances which are predicted by black box A and B differently and which are locally close to instance x. In this example, it is recognizable that the initial genetic neighborhood approach generates increased instances also for the more distant boundary areas.

Seed 3: The datasets of the third example based on our running example, shown in Figure 6.5a and 6.5b, are generated depending on the differently located yellow marked instances x. Both instances are located at: $x_1 = 55.61$ and $x_2 = -24.24$. The datasets for the binary and multiclass classifier are generated independently of each other. That means the instances (data points) of the datasets shown in the plots are not identical. For comparison, we show the generated datasets of the initial genetic neighborhood approach in Figures 6.5c and 6.5d. The dataset for the binary diff-classifier contains 1,712 instances. 1,177 instances are assigned to the class “0” (“no diff”), and 535 instances are assigned to the class “1” (“diff”). In this example, the approach does not generate a balanced dataset. As explained before, the final set Z and the instances it contains are checked for uniqueness. In this case, the genetic algorithm approach generates more redundant instances for class “1” (“diff”). But, those redundant instances are deleted. This shows that even the modified approach cannot guarantee to generate a balanced dataset in all cases. But, before the generated instances are checked for uniqueness and cleaned up, the instance generation (in the case of the binary difference classification) is balanced. The unbalanced dataset for the multiclass diff-classifier contains 1,589 instances, whereby 688 instances are classified into class “00”, 471 instances are classified into class “11”, 406 instances are classified into class “10”, and 24 instances are classified into class “01”. Figure 6.5a shows that around the center and close to x increased instances are generated.



(a) Dataset for training the binary diff-classifier

(b) Dataset for training the multiclass diff-classifier

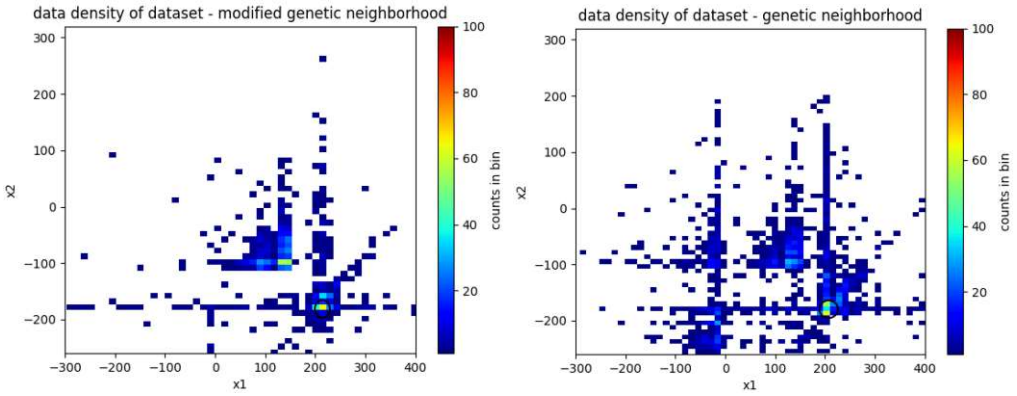
(c) Dataset for training the binary diff-classifier

(d) Dataset for training the multiclass diff-classifier

Figure 6.3: Running example: Datasets generated by the genetic neighborhood approaches with instance to be recognized in the right bottom corner (Seed 2) for training a binary and a multiclass diff-classifier to detect local differences close to instance x between black box A and B (Figure 3.5 shows the datasets of the black boxes). Depending on the yellow-marked instance, the local modified genetic neighborhood approach (see Algorithm 6.1) of DiRo2C generates the shown dataset.

Thus, increased instances are created around the boundaries to distinguish the adjacent classes (areas) in the center. Figure 6.5b also shows that the modified approach generates increased instances close to instance x. In this way, more instances of the respective adjacent outcome classes are generated close to x. But again it is also noticeable that significantly fewer instances are existing in the more distant areas. Thus, few or no instances are generated for the more distant classes (areas). For comparing the genetic neighborhood approaches we show again in Figures 6.5c and 6.5d the datasets generated by the genetic neighborhood approach.

Figure 6.6 shows the data density plots of the generated datasets of both approaches.



(a) Density of the local modified genetic neighborhood (Seed 2)

(b) Density of the local genetic neighborhood (Seed 2)

Figure 6.4: Data density of both local genetic neighborhood approaches. Figure 6.3 shows the generated datasets of the modified approach for Seed 2. The to be recognized instances x are marked with a black circle.

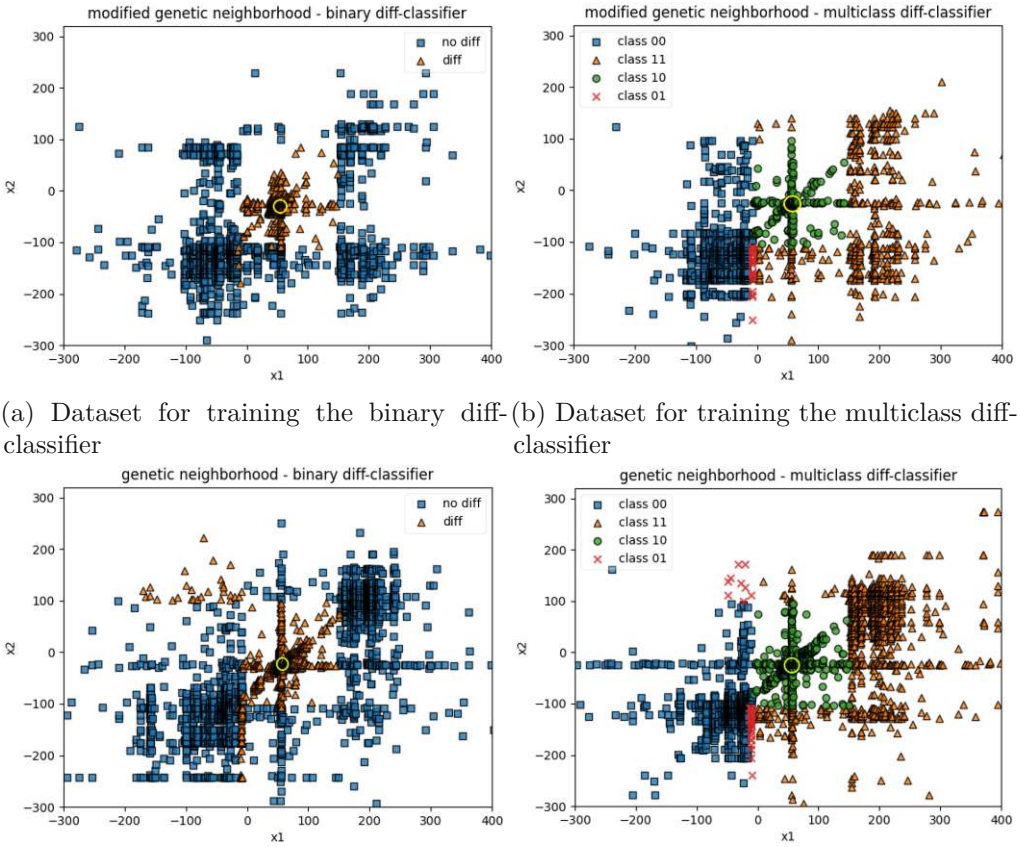
Again, we can observe the same effect of the modified genetic neighborhood approach. Both approaches generate increased instances which are predicted by black box A and B with the same result and which are locally around instance x. In contrast to the initial genetic neighborhood approach, the modified approach generates increased instances which are predicted by black box A and B differently and which are locally close to instance x. In this example, it is recognizable that the initial genetic neighborhood approach generates increased instances also for the more distant boundary areas.

We can observe that too few instances are generated in the more distant areas that the training of the diff-classifier would support to determine the boundaries more precisely. A key difference between both neighborhood approaches is that the initial neighborhood approach cannot guarantee to create increased instances which are predicted by the black boxes differently and which are as close as possible to x. The modified genetic neighborhood approach is implemented to recognize the local differences as close as possible to x. One advantage of the modified approach is the generation of a more balanced dataset in most case of the binary difference classification. After checking for uniqueness of the set Z, however, the diff-dataset may be no longer balanced.

6.2.2 Gaussian Quantiles Dataset

Now, we present another dataset example. We use two-dimensional Gaussian quantiles classification datasets[1] (see Figure 6.7) to show how the two genetic neighborhood approaches generate differently the diff-dataset to recognize the differences of black box A and B. For black box A the dataset is created with $\mu = 0$ and $\sigma^2 = 0.8$. The left dataset

[1]The datasets are available under: `https://doi.org/10.5281/zenodo.5362220`



(a) Dataset for training the binary diff-classifier

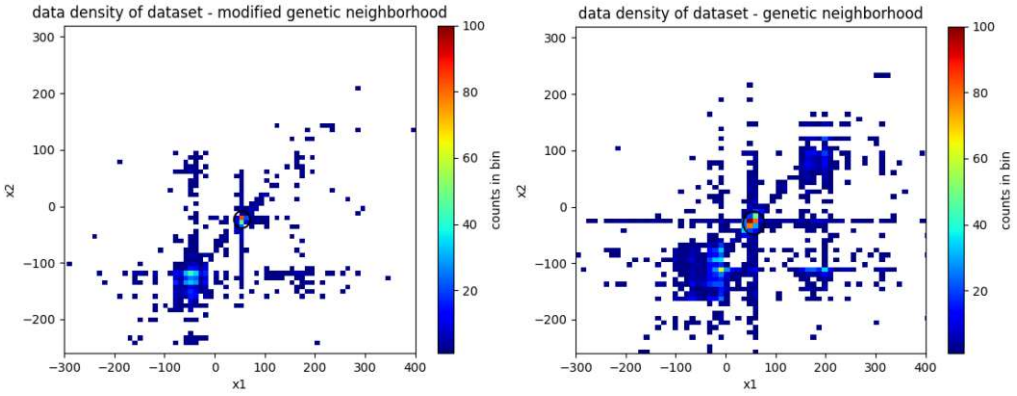
(b) Dataset for training the multiclass diff-classifier

(c) Dataset for training the binary diff-classifier

(d) Dataset for training the multiclass diff-classifier

Figure 6.5: Running example: Datasets generated by the genetic neighborhood approaches with instance to be recognized in the center (Seed 3) for training a binary and a multiclass diff-classifier to detect local differences close to instance x between black box A and B (Figure 3.5 shows the datasets of the black boxes). Depending on the yellow-marked instance, the local modified genetic neighborhood approach (see Algorithm 6.1) of DiRo2C generates the shown dataset.

for black box A shows a two-dimensional dataset with the continuous features x_1 and x_2. It contains 300 instances (data points) with the following properties for feature x_1: $min = -275.71$, $max = 255.90$, $\mu = 0.04$, and $\sigma = 88.54$ and with the following properties for feature x_2: $min = -252.57$, $max = 201.03$, $\mu = -9.44$, and $\sigma = 86.20$. The instances of the datasets are classified into two classes, "0" and "1". 150 instances of the left dataset are assigned to the class "0", and 150 instances are assigned to the class "1". The instances of the dataset for black box A are generated by the sklearn "make_gaussian_quantiles" function with the following parameters: make_gaussian_quantiles(n_samples = 300, n_classes = 2, shuffle = False, cov = 0.8, random_state = 7). Afterward, we scale the instances by the factor of 100. The manipulated dataset for black box B is generated



(a) Density of the local modified genetic neighborhood (Seed 3)

(b) Density of the local genetic neighborhood (Seed 3)

Figure 6.6: Data density of both local genetic neighborhood approaches. Figure 6.5 shows the generated datasets of the modified approach for Seed 3. The to be recognized instances x are marked with a black circle.

with $\mu = 0$ and $\sigma^2 = 1.3$. The right dataset for black box B shows a two-dimensional dataset with the continuous features x_1 and x_2. It contains 300 instances (data points) with the following properties for feature x_1: $min = -351.46$, $max = 326.21$, $\mu = 0.04$, and $\sigma = 112.87$ and with the following properties for feature x_2: $min = -321.97$, $max = 256.27$, $\mu = -12.04$, and $\sigma = 109.89$. The instances of the datasets are classified into two classes, "0" and "1". 150 instances of the right dataset are assigned to the class "0", and 150 instances are assigned to the class "1". The instances for black box B are also generated by the sklearn function but with the parameter: cov = 1.3.

The following data generation examples are based on the presented Gaussian datasets of black box A and B (see Figure 6.7). We show in the upcoming figures the resulting datasets of the various data approaches for training the classifier to recognize differences between the black boxes. So, each instance is predicted by black box A and B, and after that, the predictions are compared to assign it to the corresponding difference class. Thus, we show the resulting decision boundaries (indicated by the black drawn lines) of the trained black box A and B in Figure 3.6. Both plots show how the instances are differently predicted by black box A and B. In that example, we split the generated dataset into a training and test dataset with a ratio of 20 to 80 percent. After that, we train black box A and B using the training dataset and an SVM algorithm. In detail, we use the C-Support Vector Classification implementation of the sklearn class "sklearn.svm.SVC"[2] which is based on libsvm[3]. The plot shows the resulting decision boundaries of black box A and B. The decision boundaries represent the learned classification of the SVM model. The decision boundary runs in a circle around the center, whereby with black box B the

[2]For further details see: `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC`

[3]For further details see: `https://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf`

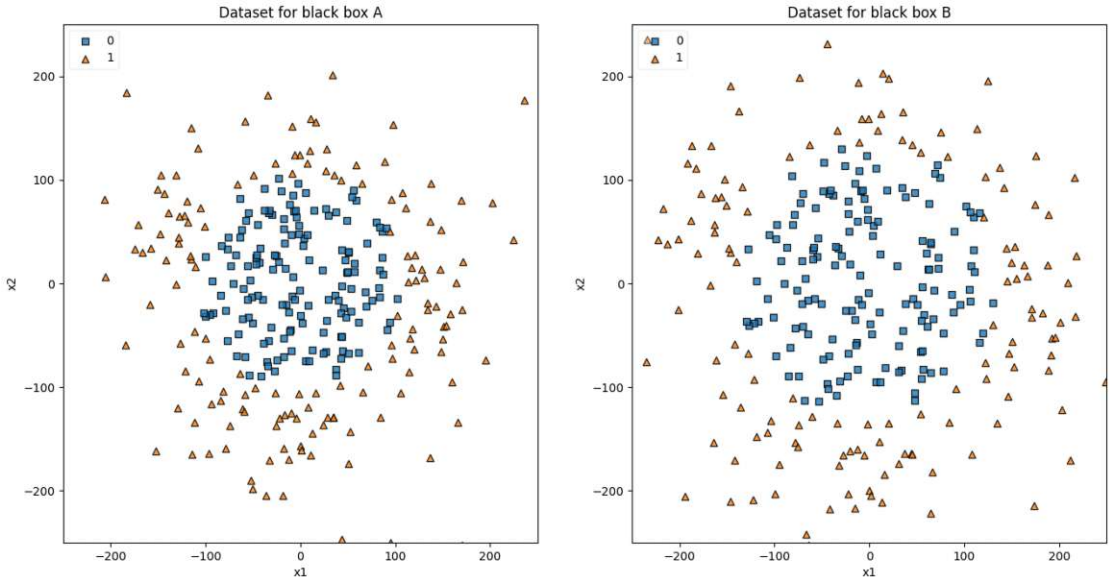


Figure 6.7: Example of a random Gaussian quantiles classification dataset for black box A and a manipulated dataset for black box B.

circle is shifted outwards. In this area enclosed by both circles (of black box A and B), the instances of the diff-dataset are classified differently.

Seed 1: For the generated diff-datasets Z_{Diff} in Figure 6.9, we set x the instance responsible for recognizing the differences as close as possible to the center (marked with a yellow circle in the figures). The instances are located at: $x_1 = -1.09$ and $x_2 = -6.77$. Figure 6.9a shows the balanced binary difference-dataset generated by the modified genetic neighborhood approach: 431 instances with $y = 0$ ("no diff") and 495 instances with $y = 1$ ("diff"). For comparison, Figure 6.9c shows the unbalanced binary difference-dataset generated by the initial genetic neighborhood approach: 2,439 instances with $y = 0$ ("no diff") and 366 instances with $y = 1$ ("diff"). Figure 6.9b shows the unbalanced multiclass difference-dataset generated by the modified genetic neighborhood approach which contains 822 instances, whereby 386 instances are classified into class "00", 40 instances are classified into class "11", 396 instances are classified into class "10", and 0 instances are classified into class "01". Figure 6.9d shows the unbalanced multiclass difference-dataset generated by the initial genetic neighborhood approach which contains 3,050 instances, whereby 813 instances are classified into class "00", 1,983 instances are classified into class "11", 254 instances are classified into class "10", and 0 instances are classified into class "01". Figure 6.10 shows the data density plots of both approaches for comparison. Again, we can observe that the modified approach generates, in the case of the binary classification, a balanced difference-dataset. We can also observe that the modified approach generates increased instances in the lower-left area of the "difference circle" to recognize the local differences as close as possible to instance x. In contrast,

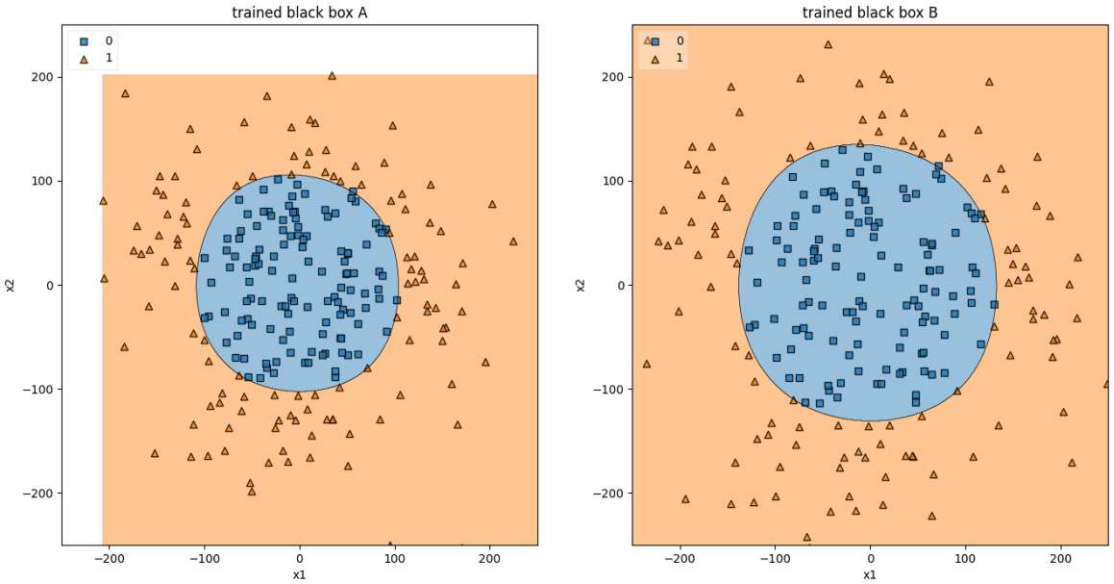
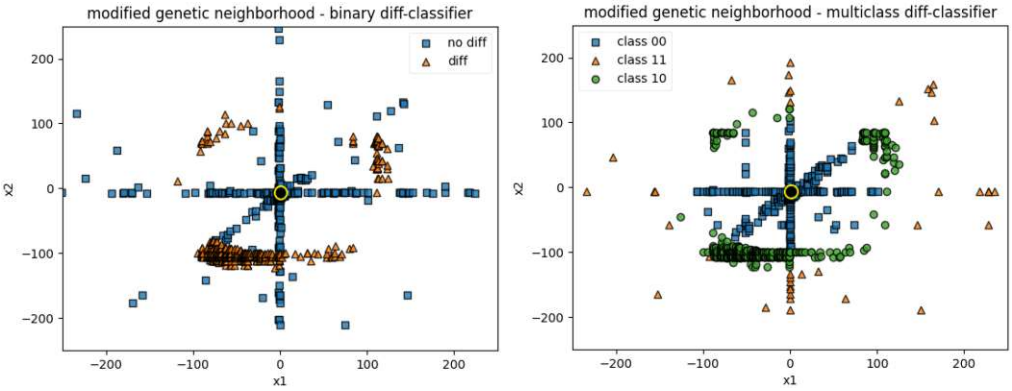


Figure 6.8: Trained SVM black box classifier A and B based on the Gaussian quantiles dataset with the corresponding decision boundaries.

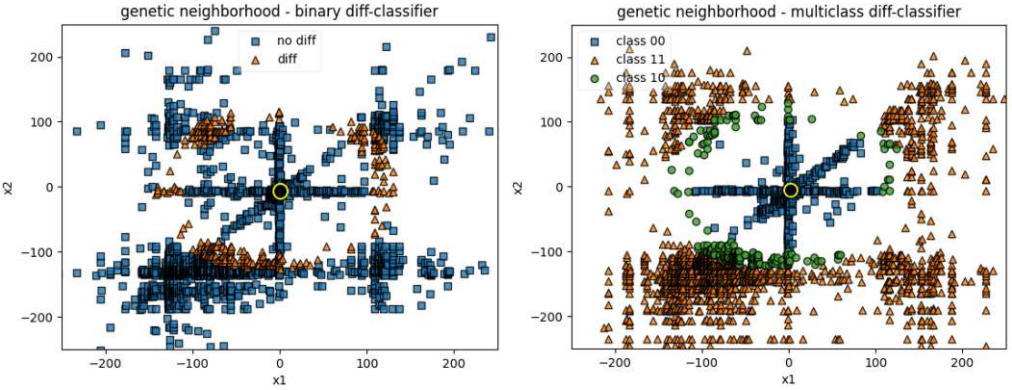
the initial genetic neighborhood approach also generates increased instances outside the "difference-circle". As we can observe in the data density plots, in this example, the initial genetic neighborhood approach generates instances in almost all areas to recognize the differences globally.

Seed 2: In the second example (see Figure 6.11) based on the Gaussian quantiles dataset, we set x the instance responsible for recognizing the differences as close as possible to the "diff" boundary in the upper-left area (marked with a yellow circle in the figures). The instances are located at: $x_1 = -101.71$ and $x_2 = -28.34$. Figure 6.11a shows the balanced binary difference-dataset generated by the modified genetic neighborhood approach: 439 instances with $y = 0$ ("no diff") and 400 instances with $y = 1$ ("diff"). For comparison, Figure 6.11c shows the unbalanced binary difference-dataset generated by the initial genetic neighborhood approach: 1,956 instances with $y = 0$ ("no diff") and 318 instances with $y = 1$ ("diff"). Figure 6.9b shows the unbalanced multiclass difference-dataset generated by the modified genetic neighborhood approach which contains 838 instances, whereby 371 instances are classified into class "00", 66 instances are classified into class "11", 401 instances are classified into class "10", and 0 instances are classified into class "01". Figure 6.9d shows the unbalanced multiclass difference-dataset generated by the initial genetic neighborhood approach which contains 2,136 instances, whereby 792 instances are classified into class "00", 1,135 instances are classified into class "11", 236 instances are classified into class "10", and 0 instances are classified into class "01". In that example, we can observe that the modified approach specifically generates increased instances in the lower-left "different-circle" area close to instance x. In contrast, the initial



(a) Dataset for training the binary diff-classifier

(b) Dataset for training the multiclass diff-classifier

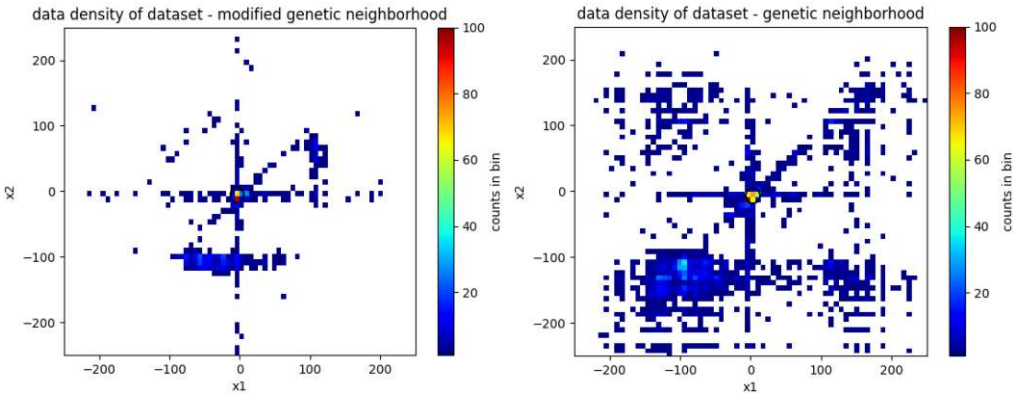


(c) Dataset for training the binary diff-classifier

(d) Dataset for training the multiclass diff-classifier

Figure 6.9: Gaussian example: Datasets generated by the genetic neighborhood approaches with instance to be recognized in the center (Seed 1) for training a binary and a multiclass diff-classifier to detect local differences close to instance x between black box A and B (Figure 6.7 shows the datasets of the black boxes). Depending on the yellow-marked instance, the local modified genetic neighborhood approach (see Algorithm 6.1) of DiRo2C generates the shown dataset.

genetic neighborhood approach generates a lot more instances which are assigned to class "no diff" (in the multiclass setting: class "00" and "11"). However, the approach does not generate evenly distributed instances within the entire "difference-circle" which are assigned to class "diff" (in the multiclass setting: class "10"). Both approaches generate increased instances within the "no diff" (in the multiclass setting: class "00" and "11") area. The main difference of the modified approach compared to the initial approach is that the modified approach focuses on one particular "diff" area and generates there increased instances close to x. The initial approach generates in all corner areas increased instances, especially for the "no diff" class(es). Figure 6.12 shows the data density plots



(a) Density of the local modified genetic neighborhood (Seed 1)

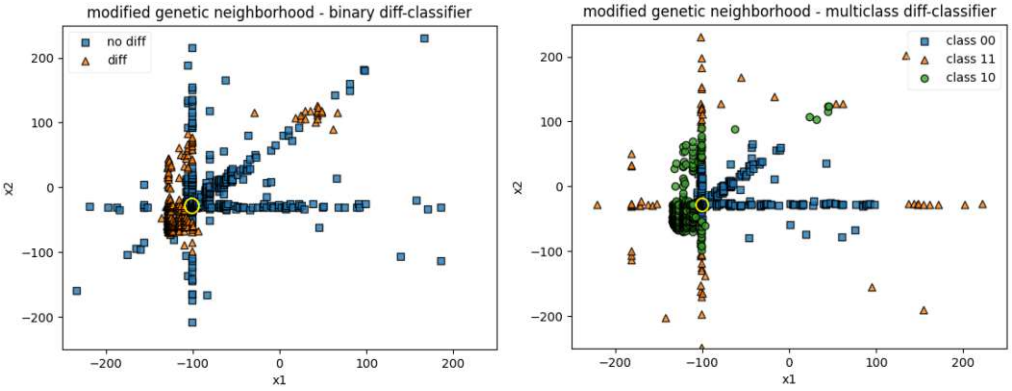
(b) Density of the local genetic neighborhood (Seed 1)

Figure 6.10: Data density of both local genetic neighborhood approaches. Figure 6.9 shows the generated datasets of the modified approach for Seed 1. The to be recognized instances x are marked with a black circle.

of both approaches for comparison.

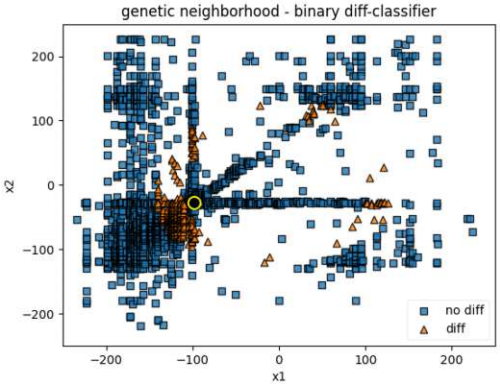
6.3 From Local to Global Explanations

By presenting the different data generation examples in the previous section, we could observe that the modified approach does not generate enough instances to detect differences in more distant areas for one particular to be recognized instance x. Thus, we can only recognize the local differences close to that particular instance x. To show the effects of the modified genetic neighborhood approach, we select different seeds for instance x manually and show only the generated dataset for one particular x. But our ultimate goal is to recognize the differences between two binary black box classifiers globally. Therefore, we already test an approach to generate various diff-datasets for differently located to be recognized instances (seeds) using the modified genetic neighborhood approach. The approach generates a defined number of to be recognized instances using a Gaussian distribution over the feature value space. After that, we concatenate the generated diff-datasets to one global diff-dataset that contains all generated instances. That global diff-dataset can be used to train a global diff-classifier that can recognize the differences between the black boxes globally. The problem is, that the current approach needs approximately 1 minute to generate one diff-dataset for one particular instance x. Assumed, we have a dataset that contains 1,000 instances and we select 10 percent of all instances and generate for each selected instance a modified genetic neighborhood. So, the generation of all genetic neighborhoods would take approximately 100 minutes. Thus, optimization of the runtime and the selection of the instances to be recognized is necessary to use our method in an efficient way. But, optimizing this process is no longer part of this thesis.

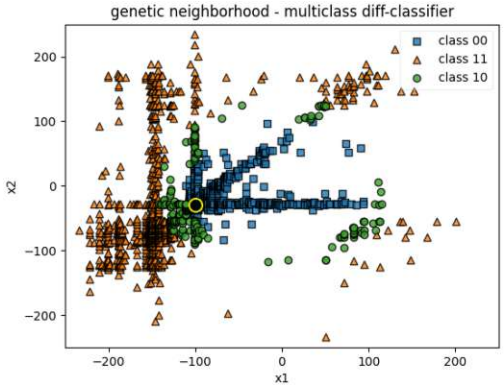


(a) Dataset for training the binary diff-classifier

(b) Dataset for training the multiclass diff-classifier



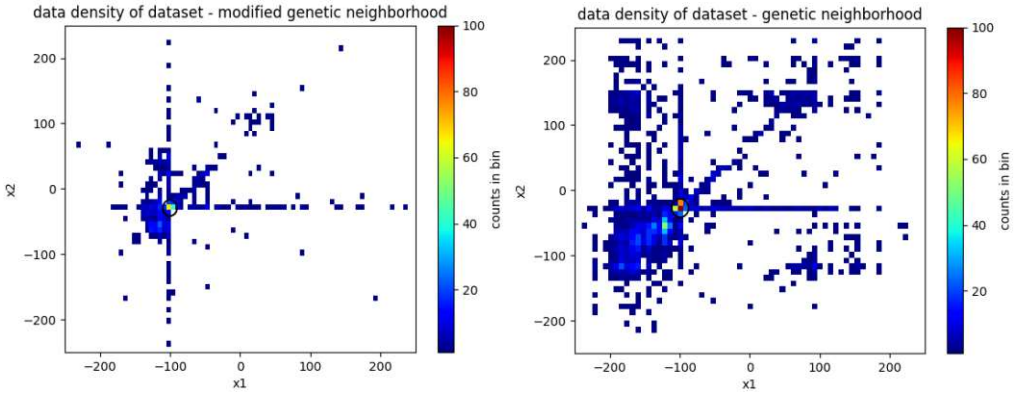
(c) Dataset for training the binary diff-classifier



(d) Dataset for training the multiclass diff-classifier

Figure 6.11: Gaussian example: Datasets generated by the genetic neighborhood approaches with instance to be recognized in the corner (Seed 2) for training a binary and a multiclass diff-classifier to detect local differences close to instance x between black box A and B (Figure 6.7 shows the datasets of the black boxes). Depending on the yellow-marked instance, the local modified genetic neighborhood approach (see Algorithm 6.1) of DiRo2C generates the shown dataset.

However, we demonstrate a global diff-dataset using our running example dataset and the modified genetic neighborhood approach for different seeds. We use the already presented seeds plus one additional seed to generate the various diff-datasets and concatenate them to one global diff-dataset. The additional to be recognized instance (Seed 4) is located at: $x_1 = -243.21$ and $x_2 = 191.26$. We select that additional seed to generate increased instances in the upper left corner to recognize the differences between black box A and B in this area. Figure 6.13 shows the global diff-dataset for the binary and the multiclass diff-classifier. The dataset for the binary diff-classifier contains 4,245 instances. 2,412 instances are assigned to the class “0” (“no diff”), and 1,833 instances are assigned to



(a) Density of the local modified genetic neighborhood Seed 2)

(b) Density of the local genetic neighborhood (Seed 2)

Figure 6.12: Data density of both local genetic neighborhood approaches. Figure 6.11 shows the generated datasets of the modified approach for Seed 2. The to be recognized instances x are marked with a black circle.

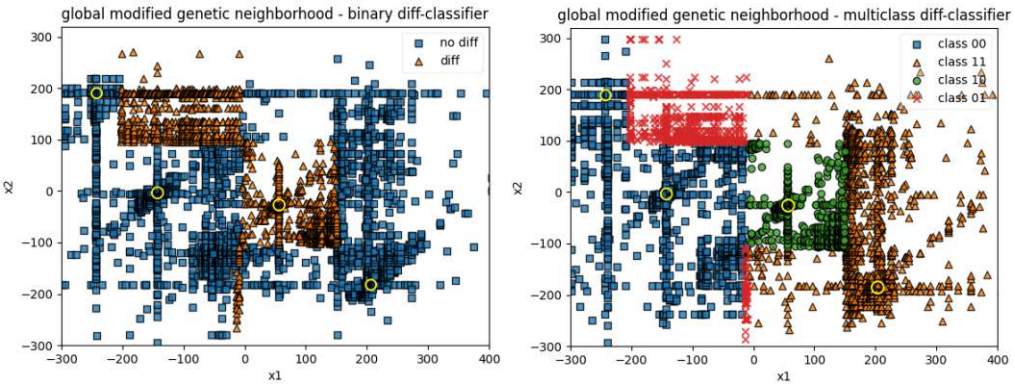
the class "1" ("diff"). The different used seeds are marked with yellow circles. The figure shows that, now, in each area where black box A and B predict different outcomes increased instances are generated close to the boundaries. Figure 6.14 shows the data density plot of the global diff-dataset. The various to be recognized instances x are marked with a black circle. We can observe that now close to that to be recognized instances increased synthetic instances are generated. So, adding various seeds that are located close to difference class boundaries helps to create a global diff-dataset that accurately covers the entire differences between black box A and B.

We will return later to the "from local to global explanations" problem in Chapter 8 to discuss potential solutions, needed optimizations, among others regarding instance selection and future work.

The following section presents how accurately the classifier, trained by using the modified approach, recognizes local differences depending on a particular instance x and why we think our approach is promising to train, finally, a diff-classifier that is able to provide global explanations.

6.4 Performance Evaluation of the Genetic Neighborhood Approaches

In this section, the final results are shown to mainly evaluate which local genetic neighborhood approach performs better to detect differences between two black box classifiers. Therefore, we again measure the already described performance metrics and present both different genetic neighborhood approaches for generating the dataset Z_{diff} to train a diff-classifier. Finally, we also discuss our major findings. To measure the performance of



(a) Global dataset for training the binary diff-classifier

(b) Global dataset for training the multiclass diff-classifier

Figure 6.13: Running example: Binary and multiclass global diff-dataset generated by using the modified genetic neighborhood approach and various instances to be recognized for training a binary and a multiclass diff-classifier to recognize the global differences between black box A and B (Figure 3.5 shows the datasets of the black boxes). Depending on the yellow-marked instances, the local modified genetic neighborhood approach (see Algorithm 6.1) of DiRo2C generates the diff-datasets and concatenate it to the shown dataset.

the modified approach, we again use the experimental setup presented in Chapter 5.

In Table 6.1, we present the measured metrics. Regarding RQ 1 and our hypothesis, the *Accuracy* results of the binary diff-classifiers are, as expected, again closer to 1 compared to the multiclass diff-classifiers measures. Furthermore, the *Accuracy* compared to the two genetic neighborhood approaches also does not differ significantly. But the F_1-*score* in the modified approach is consistently approximately 1 across all data sets and is, therefore, better than compared to the initial genetic neighborhood approach. In addition, the F_1-*scores* of the modified genetic neighborhood trained binary diff-classifiers are higher across all datasets than the F_1-*scores* of the multiclass diff-classifiers. The *PearsonCC* once again confirms the performance results. The results shown in Table 5.7 and 6.1 can be used to state the following. Concerning RQ 1.1 and 1.2, the measures show that a diff-classifier trained on synthetic data can outperform a classifier created using a real and already existing dataset. They also show that the diff-classifiers trained by datasets generated by the local modified genetic neighborhood approach perform better than the proposed global strategies. We have shown in this chapter how the modified genetic neighborhood solution generates increased data in the area where the predictions of the two black box classifiers differ. That modification finally leads to a higher data density in the immediate area close to a specific instance x and thus to higher coverage.

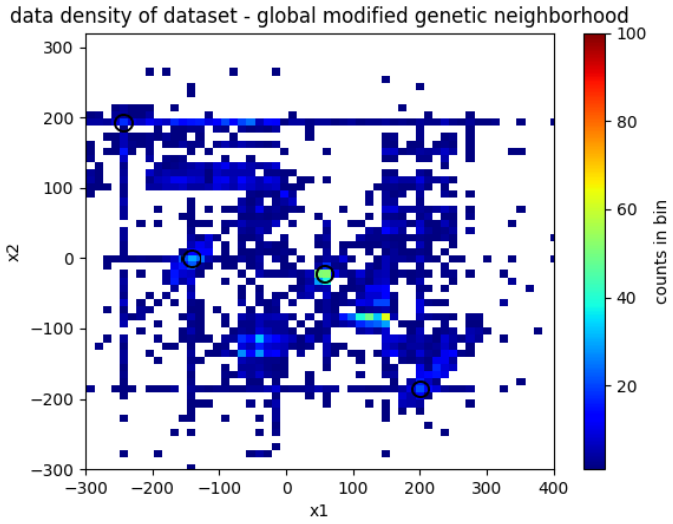


Figure 6.14: Data density of the global modified genetic neighborhood dataset based on the running example. Figure 6.13 shows the generated global datasets using the modified approach and various instances. The to be recognized instances x are marked with a black circle.

6.5 Summary

In summary, the decisive change of the modified approach is the modification of the fitness functions by comparing the outcome of black box A and B. Therefore, the modified approach should generate, finally, a more balanced diff-dataset $Zdiff$ by considering that the check for uniqueness can lead to a dataset that is no longer being balanced. We also have explained that we have presented manually picked seeds and show the effect of the local modified genetic neighborhood approach for one particular instance x. Furthermore, we have explained the "from local to global explanation" problem. Our evaluations have shown that the classifiers trained by the dataset generated with the modified approach outperform the classifiers based on the initial genetic neighborhood approach. In the following chapter, we will evaluate the detected differences for correctness. Therefore, we will show the resulting decision tree of trained diff-classifiers to analyze the trained decision rules. We will also show how our approach tackles non-orthogonal class boundaries and whether the resulting decision rules are interpretable anymore.

Dataset	DCM	DA	*ACC*	F_1-*score*	*PCC*
adult	binary dc	gn	**.984 ± .006**	.703 ± .055	.695 ± .073
		mod gn	.983 ± .013	**.982 ± .015**	**.968 ± .026**
	multiclass dc	gn	**.906 ± .045**	.667 ± .039	.814 ± .089
		mod gn	.905 ± .062	**.753 ± .108**	**.844 ± .090**
bank marketing	binary dc	gn	**.996 ± .003**	.711 ± .357	.717 ± .361
		mod gn	.991 ± .007	**.987 ± .010**	**.982 ± .014**
	multiclass dc	gn	.931 ± .015	.759 ± .126	.814 ± .089
		mod gn	**.953 ± .021**	**.888 ± .032**	**.916 ± .034**
credit approval	binary dc	gn	**.983 ± .009**	.911 ± .058	.900 ± .044
		mod gn	.981 ± .014	**.980 ± .015**	**.962 ± .029**
	multiclass dc	gn	**.979 ± .012**	.897 ± .050	**.966 ± .015**
		mod gn	.970 ± .015	**.926 ± .026**	.951 ± .025

Table 6.1: Genetic neighborhood approaches: comparison of the diff-classifier performance results. The following abbreviations are used: DCM (Diff-Classifier Method), DA (Data Approach), ACC (Accuracy), *PCC* (Pearson CC), and the abbrevations for the data generation approaches gn (local genetic neighborhood), and mod gn (modified local genetic neighborhood).

CHAPTER 7

Evaluation of the Detected Differences

So far, we have already presented our measures regarding RQ 1 (and sub-questions). Next, we will focus on showing our results concerning RQ 2. To evaluate if the trained decision tree-based diff-classifier recognizes the actual (true) differences, we manipulate particular features and verify if the forced differences are found in the resulting decision tree. We use different settings to show our results. At first, we present our results by using two-dimensional synthetic datasets. We again use our *running example* and the *diagonal example*. In the case of the *diagonal example*, we use for black box B a diagonal boundary to show the effect of a non-orthogonal difference-class boundary between black box A and B. We use the *diagonal example* already to explain the difference classification problem in Chapter 3 (see Figure 3.2). Second, we present our results regarding the multi-dimensional *adult* dataset. We run different data manipulation experiments to force differences between black boxes A and B.

7.1 Synthetic Datasets

In this section, we present the trained diff-classifiers based on two-dimensional synthetic datasets. We use our *running example* setting and an additional *diagonal example* setting. Using the two-dimensional synthetic datasets, we can better illustrate the resulting decision boundaries of the trained diff-classifier.

7.1.1 Running Example Dataset

Next, we show our results based on our running example dataset. We want to mention that the *global synthetic data*, *local genetic neighborhood data*, and *local modified neighborhood*

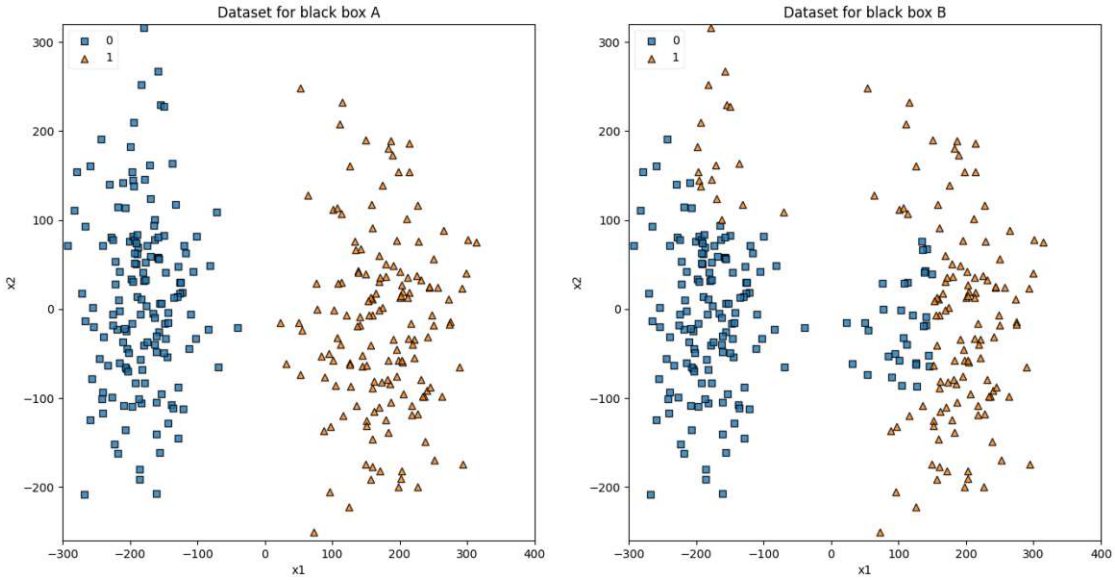


Figure 7.1: Running example of a synthetic classification dataset for black box A and a manipulated dataset for black box B. The plot of the dataset of black box B shows the different shifted boundaries of the class membership.

data approach generates the instances with no specified random seed[1], which means the instances of the datasets shown in the plots are not identical. In addition, the presented datasets are not identical compared to the already presented datasets in earlier chapters. At first, once again, we show the dataset of black box A and B in Figure 7.1. We also want to show the decision boundaries of the trained black box classifier A and B using a CART algorithm in Figure 7.2. And additionally, to compare the different bases for decision-making of the two black boxes in detail, Figure 7.3a again shows the decision tree of black box A and Figure 7.3b shows the decision tree of black box B.

Next, we show the decision boundaries of the trained diff-classifiers for the various data approaches based on the datasets, as shown in Figure 7.1. For the local data approaches, we use the same to be recognized instance x located at: $x_1 = -143.91$ and $x_2 = -4.53$ (Seed 1). For the local data approaches, we also show the trained decision boundaries of a global diff-classifier trained by a global dataset where the diff-datasets for the already presented 4 various seeds (cf. Section 6.3) are concatenated.

Global Synthetic Data

Figure 7.4 shows the decision boundaries of the trained diff-classifier based on the global synthetic data approach. Both diff-classifiers (the binary classifier and the multiclass

[1]The implementations of the genetic approaches are based on the implementation of LORE. In the implementation of LORE, the genetic neighborhoods are also generated without a defined random seed. Furthermore, we have not changed the initialization of the genetic algorithm.

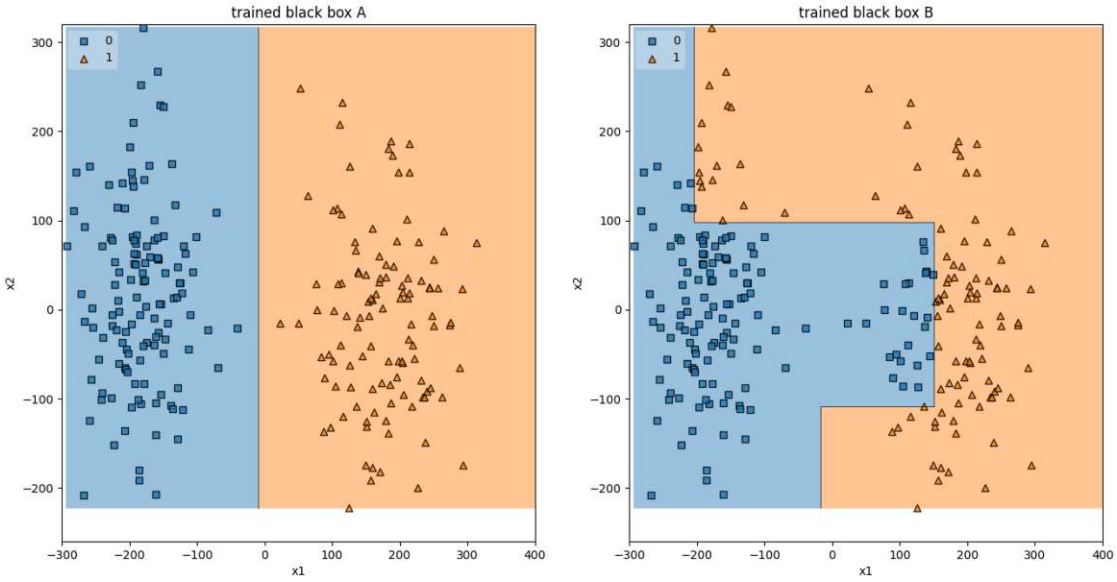


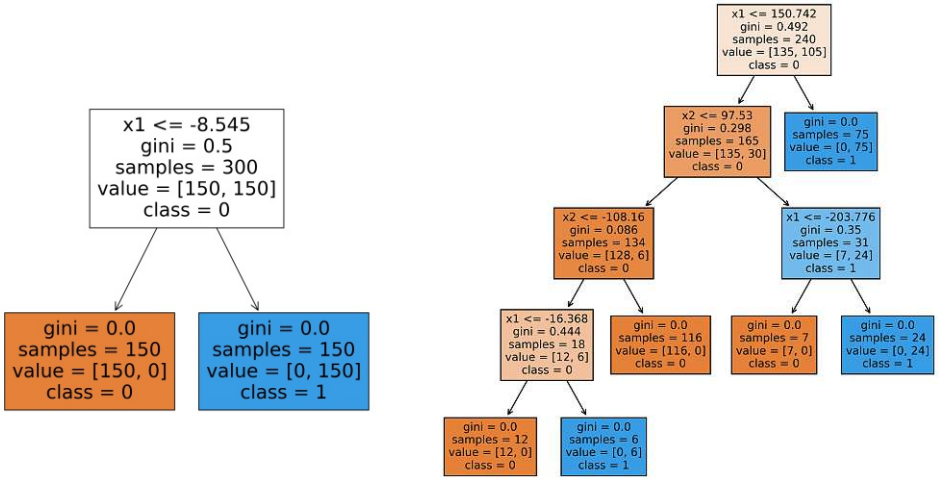
Figure 7.2: Trained black box classifier A and B with the corresponding decision boundaries based on the running example dataset.

classifier) recognize the global differences between black boxes A and B. In this two-dimensional setting, the approach delivers accurate decision boundaries. The approach recognizes all manipulated areas, also the narrow manipulated area in the middle below. But, in the case of complex datasets such as the *adult* dataset, this approach yields poor results in terms of performance (cf. Chapter 5).

In addition, we also want to show the decision trees of the classifiers (see Figure 7.5). The conditions for the classification are shown here in detail.

Global Real Data

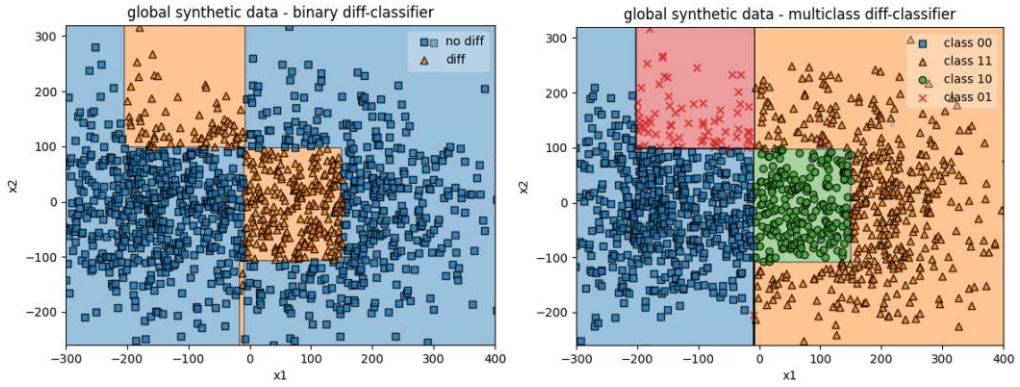
Figure 7.6 shows the decision boundaries of the trained diff-classifier based on the global real data approach. Both diff-classifiers (the binary classifier and the multiclass classifier) recognize not overall the global differences between black boxes A and B. The actual differences in the middle-lower area are not recognized. In the upper left (red) area, which is dedicated to the difference-class "01", the decision boundaries could be determined almost exactly, whereby there are almost no instances close to the right boundary (to class "11"). But, in the middle area that is dedicated to the difference-class "10", the decision boundaries are not recognized exactly. In that area, the trained decision boundaries are shifted compared to the actual boundaries. In this two-dimensional setting, the approach delivers in many areas accurate decision boundaries. However, in the case of complex benchmark datasets such as the *adult* dataset, this approach also yields poor results in terms of performance (cf. Chapter 5).



(a) Decision tree of black box classifier A

(b) Decision tree of manipulated black box classifier B

Figure 7.3: Decision tree of the trained binary black box classifier A and manipulated binary black box classifier B (Figure 3.5 shows the datasets of the black boxes). If the condition for a particular node is met, the path on the left of the decision tree is chosen.



(a) Decision boundaries of binary diff-classifier

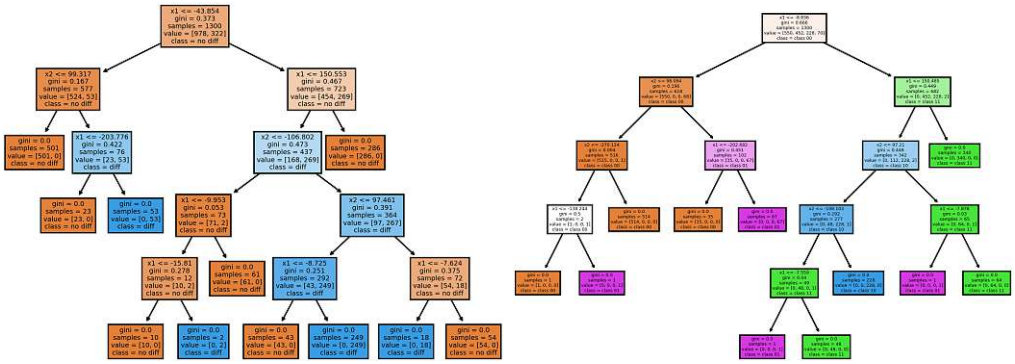
(b) Decision boundaries of multiclass diff-classifier

Figure 7.4: Diff-classifiers trained by the global synthetic data approach based on the running example dataset.

In addition, we also want to show the decision trees of the classifiers (see Figure 7.7). The conditions for the classification are shown here in detail.

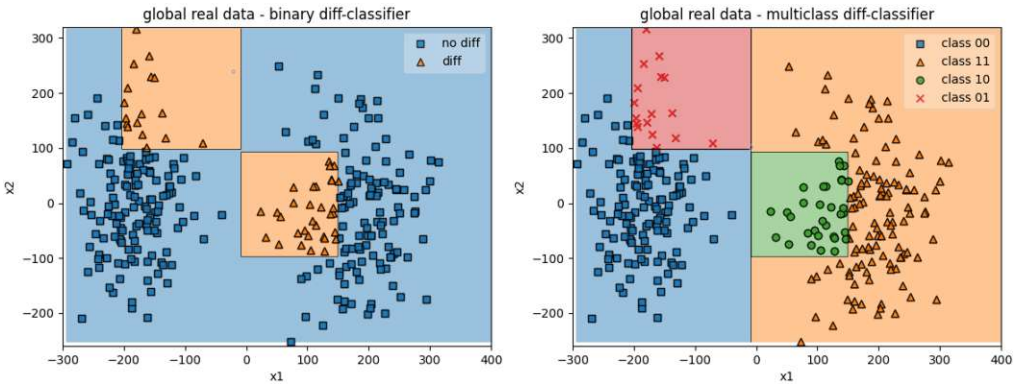
Local Real Data

For the local real data approach, we present at first the decision boundaries of the trained diff-classifier for the manually defined Seed 1.



(a) Decision tree of binary diff-classifier (b) Decision tree of multiclass diff-classifier

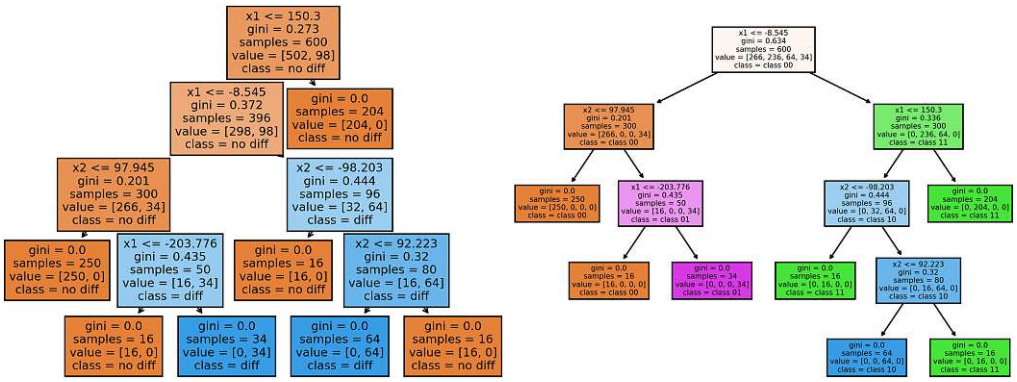
Figure 7.5: Decision trees of the diff-classifiers trained by the global synthetic data approach based on the running example dataset.



(a) Decision boundaries of binary diff-classifier (b) Decision boundaries of multiclass diff-classifier

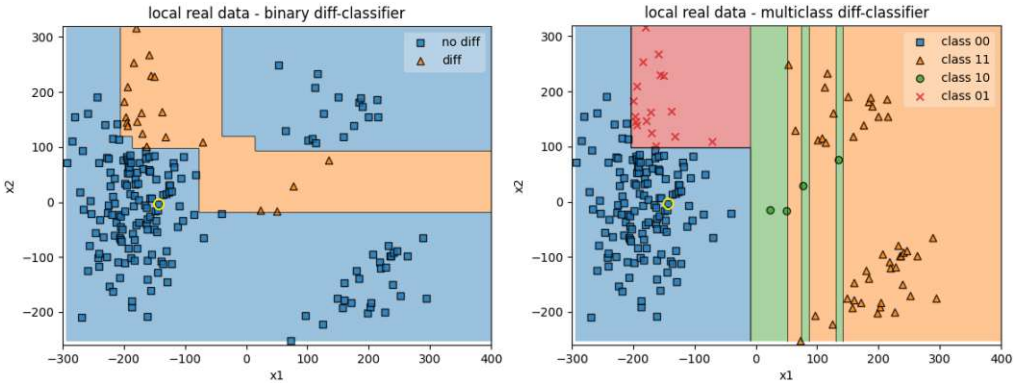
Figure 7.6: Diff-classifiers trained by the global real data approach based on the running example dataset.

Seed 1: Figure 7.8 shows the decision boundaries of the trained diff-classifiers based on the local real data approach. Each dataset is generated based on the instance x highlighted in yellow, and both instances are located at: $x_1 = -143.91$ and $x_2 = -4.53$. At first, we discuss the resulting decision boundaries of the binary diff-classifier. Figure 7.8a shows the decision boundaries of the binary diff-classifier. It shows that in the left area ($x_1 < 8.5$), the diff-classifier can classify the differences accurately. In this approach, however, the classifier is dependent on the existing instances of the real dataset. But in the more distant (right) areas, the diff-classifier cannot recognize the actual differences. Figure 7.8b shows the decision boundaries of the multiclass diff-classifier. In the case of the multiclass setting, we also can see that the differences can be recognized accurately in the left area ($x_1 < -8.5$). But in the more right area, the trained decision boundaries



(a) Decision tree of binary diff-classifier (b) Decision tree of multiclass diff-classifier

Figure 7.7: Decision trees of the diff-classifiers trained by the global real data approach based on the running example dataset.



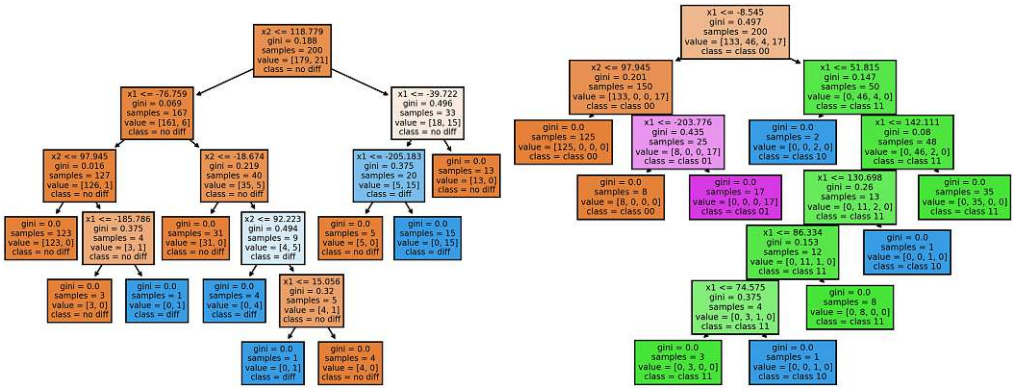
(a) Decision boundaries of binary diff-classifier (b) Decision boundaries of multiclass diff-classifier

Figure 7.8: Diff-classifiers trained by the local real data approach based on the running example dataset (Seed 1).

do not reflect the actual differences between black box A and B. In addition, we also want to show the decision trees of the classifiers (see Figure 7.9). The conditions for the classification are shown here in detail.

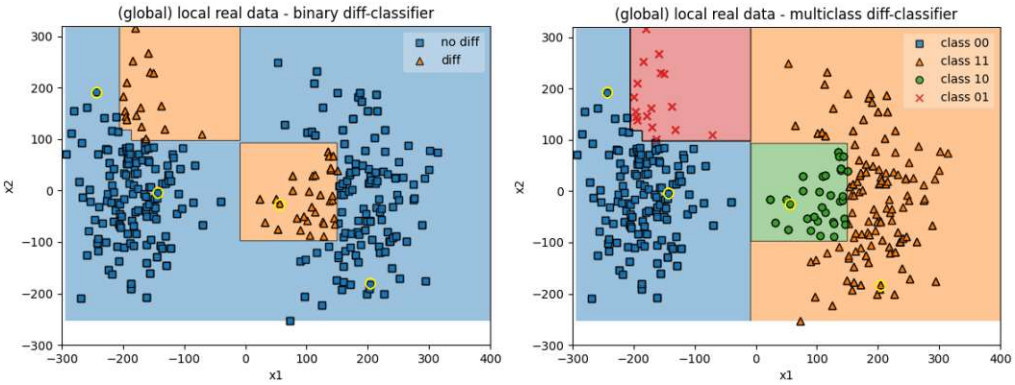
Next, we present the results for a global binary and multiclass diff-classifier. We use a concatenated global diff-dataset based on the presented 4 different located seeds. That means we use the local real data approach for that 4 various located instances to be recognized and merge the dataset to a global diff-dataset and train the classifier.

Global local real data: Figure 7.10 shows the decision boundaries of the trained global diff-classifiers based on the local real data approach. The merged global dataset



(a) Decision tree of binary diff-classifier (b) Decision tree of multiclass diff-classifier

Figure 7.9: Decision trees of the diff-classifiers trained by the local real data approach based on the running example dataset (Seed 1).

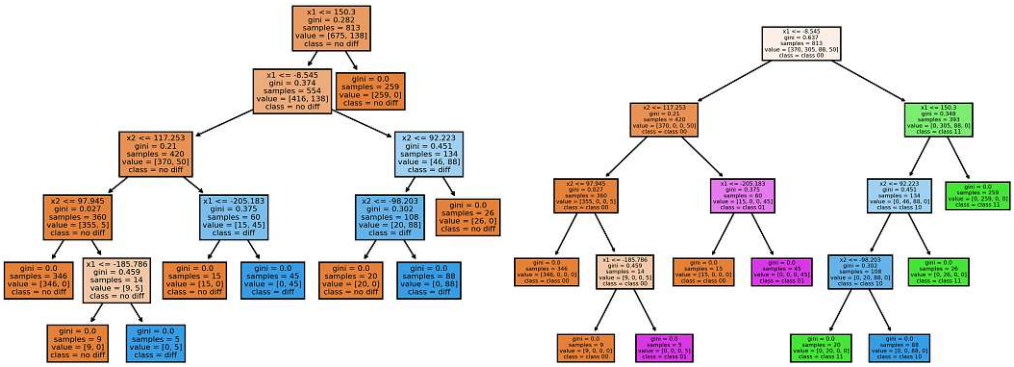


(a) Decision boundaries of binary diff-classifier (b) Decision boundaries of multiclass diff-classifier

Figure 7.10: Global diff-classifiers trained by the global local real data approach based on the running example dataset and various seeds.

is generated based on the instances highlighted in yellow. In that case, the global diff-classifier can find the differences for each area accurately (except the middle-lower area). But in principle, it is not an improvement compared to the previously presented global real data approach. That means the more datasets for different instances are concatenated to a global dataset, the more instances the dataset contains from the existing dataset of the black box models.

In addition, we also want to show the decision trees of the classifiers (see Figure 7.11). The conditions for the classification are shown here in detail.



(a) Decision tree of binary diff-classifier (b) Decision tree of multiclass diff-classifier

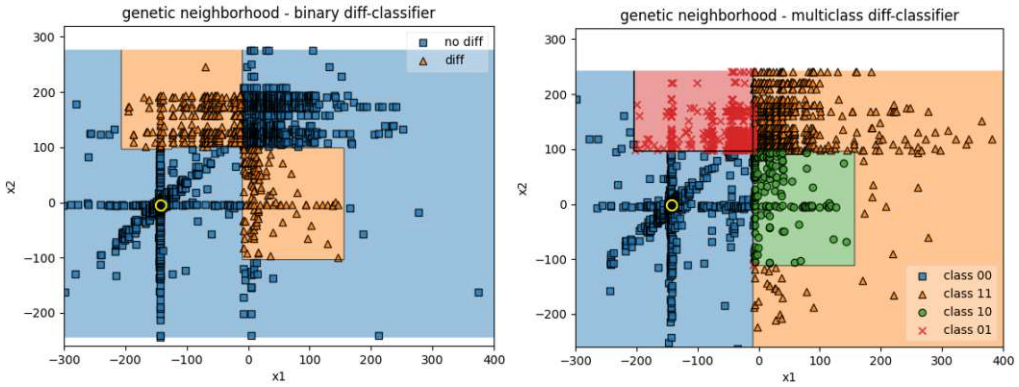
Figure 7.11: Decision trees of the diff-classifiers trained by the global local real data approach based on the running example dataset.

Local Genetic Neighborhood Data

For the local genetic neighborhood data approach, we also present at first the trained decision boundaries of the defined Seed 1.

Seed 1: Figure 7.12 shows the decision boundaries of the trained diff-classifiers based on the genetic neighborhood data approach. Each dataset is generated based on the instance x highlighted in yellow, and both instances are located at: $x_1 = -143.91$ and $x_2 = -4.53$. At first, we discuss the resulting decision boundaries of the binary diff-classifier. Figure 7.12a shows the decision boundaries of the binary diff-classifier. The trained diff-classifier recognizes nearly all difference-areas, except the middle-lower "diff" area, because in that area, too few instances are generated to recognize the actual differences. The upper and right, close to instance x, decision boundaries can be recognized accurately through increased generated instances along the boundaries. We also can observe that the more distant boundaries can be recognized almost exactly. But for comparison, not that many instances are generated in the more distant areas (to be seen in the right part of the middle "diff" area). Figure 7.12b shows the decision boundaries of the multiclass diff-classifier. We also can observe the same behavior in the multiclass setting. In summary, the genetic algorithm creates no instances in the lower middle area, where black box A classifies the "0" class, and black box B classifies the "1" class. In addition, we also want to show the decision trees of the classifiers (see Figure 7.13). The conditions for the classification are shown here in detail.

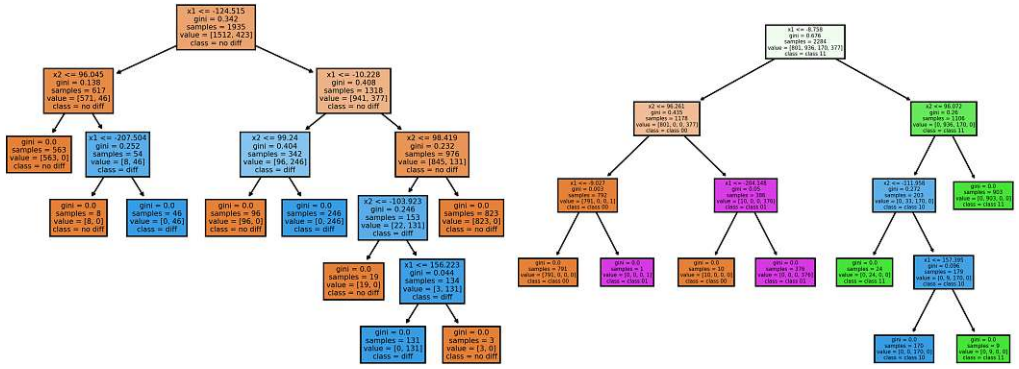
Next, again, we present the results for a global binary and multiclass diff-classifier. We use a concatenated global diff-dataset based on the presented 4 different located seeds. That means we use the local real data approach for that 4 various located instances to be recognized and merge the dataset to a global diff-dataset and train the classifier.



(a) Decision boundaries of binary diff-classifier

(b) Decision boundaries of multiclass diff-classifier

Figure 7.12: Diff-classifiers trained by the local genetic neighborhood data approach based on the running example dataset (Seed 1).

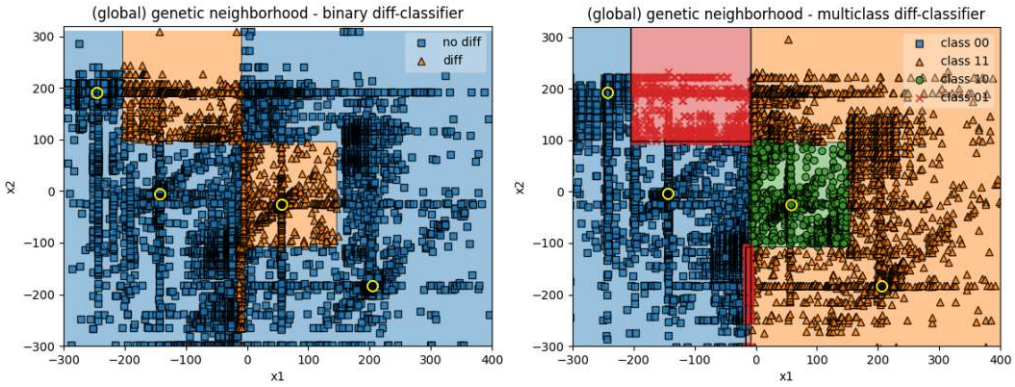


(a) Decision tree of binary diff-classifier

(b) Decision tree of multiclass diff-classifier

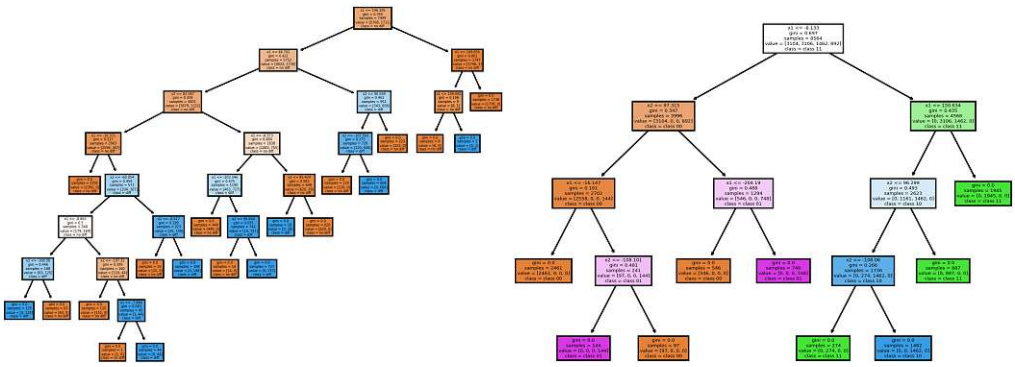
Figure 7.13: Decision trees of the diff-classifiers trained by the modified genetic neighborhood data approach based on the running example dataset (Seed 1).

Global Genetic Neighborhood: Figure 7.14 shows the decision boundaries of the trained global diff-classifiers based on the concatenated genetic neighborhood diff-datasets. Each dataset is generated based on the instances highlighted in yellow. The unbalanced binary diff-dataset contains 7,499 instances, whereby 5,768 instances are classified into class “no diff” and 1,731 instances are classified into class “diff”. The multiclass diff-dataset contains 8,564 instances, whereby 3,104 instances are classified into class “00”, 3,106 instances are classified into class “11”, 1,462 instances are classified into class “10”, and 892 instances are classified into class “01”. At first, we discuss the resulting decision boundaries of the binary diff-classifier. Figure 7.14a shows the decision boundaries of the binary diff-classifier. Now, the global approach generates in all areas (where difference transitions between black box A and B exist) sufficient enough instances to recognize the



(a) Decision boundaries of binary diff-classifier (b) Decision boundaries of multiclass diff-classifier

Figure 7.14: Global diff-classifiers trained by the concatenated local genetic neighborhood diff-datasets based on the running example dataset.

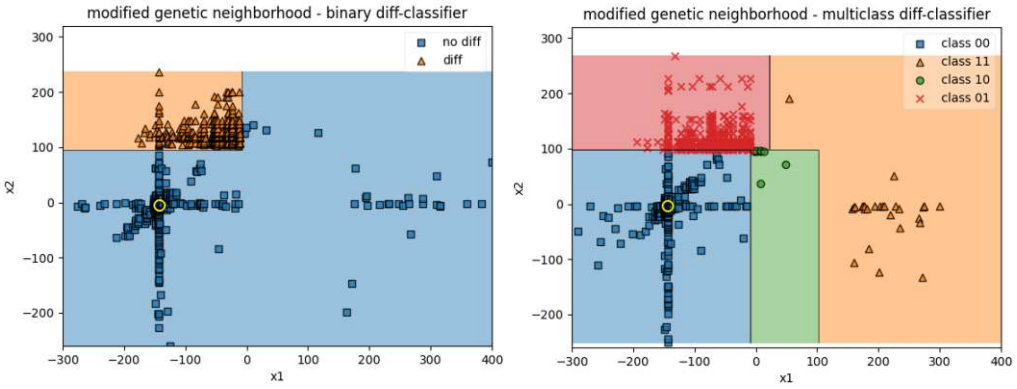


(a) Decision tree of binary diff-classifier (b) Decision tree of multiclass diff-classifier

Figure 7.15: Decision trees of the global diff-classifiers trained by the concatenated genetic neighborhood diff-datasets based on the running example dataset.

decision boundaries reliable. Figure 7.14b shows the decision boundaries of the multiclass diff-classifier. We can also observe that the global differences are recognized accurately. Furthermore, we can observe that the more instances to be recognized are selected, the more instances are generated to recognize the global differences. However, when evaluating the performance of the diff-classifier based on the three complex benchmark datasets, we can see that the simplistic genetic neighborhood approach performs worse than the modified approach. In addition, we also want to show the decision trees of the classifiers (see Figure 7.15). The conditions for the classification are shown here in detail.

Next, we show the detected differences of the diff-classifiers based on the local modified genetic neighborhood data approach.



(a) Decision boundaries of binary diff-classifier

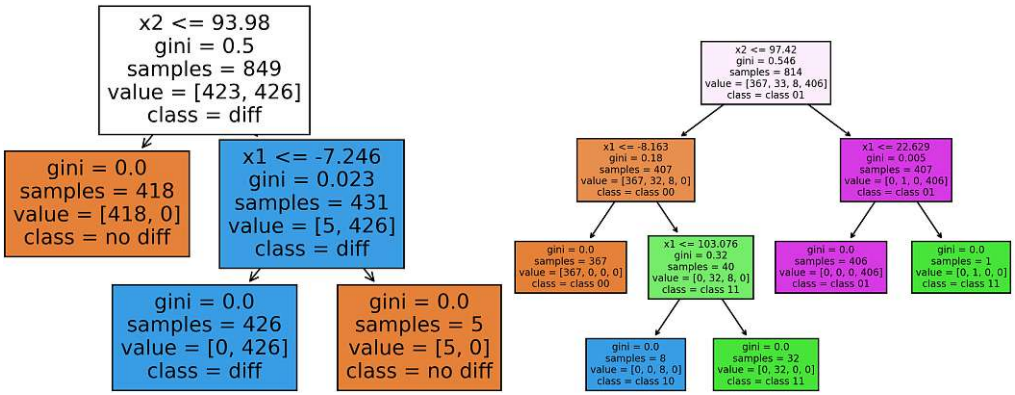
(b) Decision boundaries of multiclass diff-classifier

Figure 7.16: Diff-classifiers trained by the local modified genetic neighborhood data approach based on the running example dataset (for Seed 1).

Local Modified Genetic Neighborhood Data

For the local modified genetic neighborhood data approach, we also present at first the trained decision boundaries of the defined Seed 1.

Seed 1: Figure 7.16 shows the decision boundaries of the trained diff-classifiers based on the modified genetic neighborhood data approach. Each dataset is generated based on the instance x highlighted in yellow, and both instances are located at: $x_1 = -143.91$ and $x_2 = -4.53$. At first, we discuss the resulting decision boundaries of the binary diff-classifier. Figure 7.16a shows the decision boundaries of the binary diff-classifier. We can confirm that the approach generates in the top left, middle and middle-lower area too few instances to train rules to be able to differentiate between class “no diff” and “diff”. But, the approach generates increased instances in the locally closer top “diff” area. We also can confirm that in all of the more distant areas, few or no instances are generated to distinguish between the difference-classes. Figure 7.16b shows the decision boundaries of the multiclass diff-classifier. We can also see here that the differences are not recognized in the areas mentioned before. Therefore, we also can confirm, increased instances are generated to recognize the locally closer decision boundary in the top area between the difference-class “00” and “01”. But not in the top left area. Only a few instances are generated in the middle area where the difference-class “00” borders the difference-class“10”. No instances are generated in the middle-lower area where the difference-class “00” borders the class“01”. As a result, an unbalanced multiclass diff-dataset is generated. The dataset contains only a few generated instances that are assigned to the difference-classes “10” and “11”. In addition, we also want to show the decision trees of the classifiers (see Figure 7.17). The conditions for the classification are shown here in detail.



(a) Decision tree of binary diff-classifier (b) Decision tree of multiclass diff-classifier

Figure 7.17: Decision trees of the diff-classifiers trained by the modified genetic neighborhood data approach based on the running example dataset (for Seed 1).

Listing 7.1: Decision rules - binary diff-classifier (Seed 1)

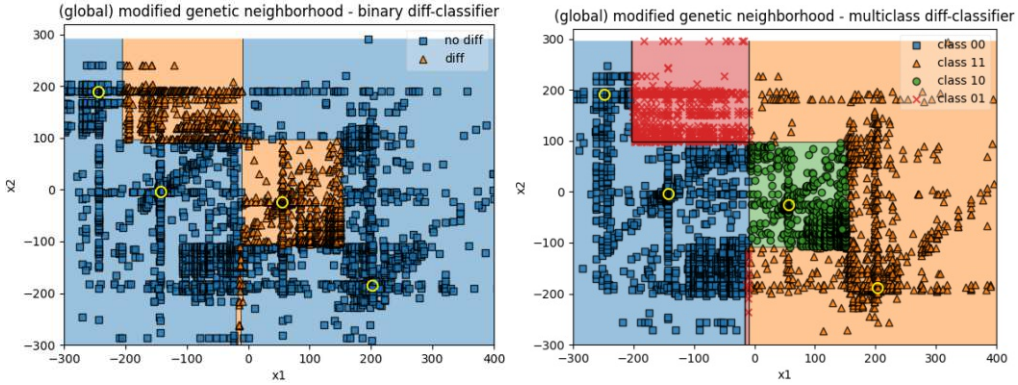
```
{'diff': '{x2 > 93.98, x1 <= −7.25}'}
```

Listing 7.2: Decision rules - multiclass diff-classifier (Seed 1)

```
{'10': '{x2 <= 97.42, x1 > −8.163, x1 <= 103.08}'},
{'01': '{x2 > 97.42, x1 <= 22.63}'}
```

For the modified genetic neighborhood data approach, we present the derived decision rules. For each rule, the difference-class to which the conditions apply is displayed first and then the logical conjunction of conditions (separated by commas). Listing 7.1 shows the decision rules derived from the binary decision tree (see Figure 7.17a). Each set of rules represents a path of the decision tree from the root to a "diff" leaf. We only show the "diff" decision rule paths. Listing 7.2 shows the decision rules derived from the multiclass decision tree (see Figure 7.17b). Each set of rules represents a path of the decision tree from the root to a "01" or "10" diff-leaf. Again, we only show the "diff" decision rule paths.

In that case, the decision rules are comprehensible and can be used to explain the differences compactly. The problem is that the differences are only accurately recognized locally close to x. The recognized differences and the decision rules are therefore not applicable for global explanations. However, the ultimate goal is to recognize the global differences between black box A and B. Thus, we present again the results for the trained binary and multiclass diff-classifier using a concatenated global diff-dataset based on the 4 different located seeds.



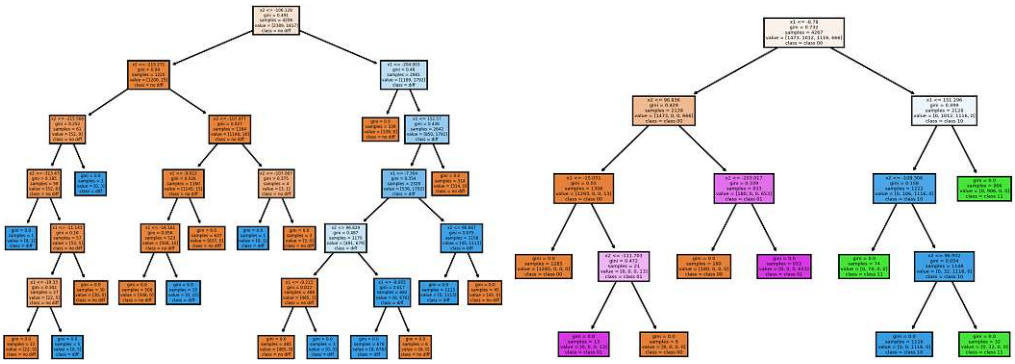
(a) Decision boundaries of binary diff-classifier

(b) Decision boundaries of multiclass diff-classifier

Figure 7.18: Global diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the running example dataset.

Global Modified Genetic Neighborhood: Figure 7.18 shows the decision boundaries of the trained global diff-classifiers based on the concatenated modified genetic neighborhood diff-datasets. Each dataset is generated based on the instances highlighted in yellow. At first, we discuss the resulting decision boundaries of the binary diff-classifier. Figure 7.18a shows the decision boundaries of the binary diff-classifier. Now, we can observe that the global approach generates instances to recognize the decision boundaries in all areas, where "difference-class" transitions between black boxes A and B exist. But compared to the initial genetic neighborhood approach, not that many instances are generated overall. The reason for this is as follows: The initial genetic neighborhood approach generates genetic neighborhoods for both black boxes A and B independently of each other. However, the modified approach generates only one genetic neighborhood, dependent on each other. Figure 7.18b shows the decision boundaries of the multiclass diff-classifier. Here, we can observe the same effect that not as many instances are generated overall. Generally, we can observe, the more instances to be recognized are selected, the more instances are generated to recognize the global differences reliable. We want to mention that an optimized approach is required to select the instances to be recognized. However, this is out of scope for this thesis. In addition, we also want to show the decision trees of the classifiers (see Figure 7.19). The conditions for the classification are shown here in detail.

Listing 7.3 shows the decision rules derived from the binary decision tree (see Figure 7.19a) to explain the global decision differences between black box A and B. Each set of rules represents a path of the decision tree from the root to a "diff" leaf. We only show the "diff" decision rule paths. But, in the case of the binary setting, we can observe that the rules can become very complex, even for a simple two-dimensional dataset. Compared to the decision tree and the rules of the global diff-classifier based on the initial genetic neighborhood approach, we can observe that the rules are more complex than necessary.



(a) Decision tree of binary diff-classifier (b) Decision tree of multiclass diff-classifier

Figure 7.19: Decision trees of the global diff-classifiers trained by the concatenated modified genetic neighborhood diff-datasets based on the running example dataset.

Listing 7.3: Decision rules - global binary diff-classifier

```
{'diff': '{x2 <= -106.18, x2 <= -215.27, x2 <= -215.57, x2 <=
    -313.47}'},
{'diff': '{x2 <= -106.13, x2 <= -215.27, x2 <= -215.57, x2 >
    -313.47, x1 <= -11.14, x1 > -19.33}'},
{'diff': '{x2 <= -106.13, x2 <= -215.27, x2 > -215.57}'},
{'diff': '{x2 <= -106.13, x2 > -215.27, x2 <= -107.98, x1 <=
    -9.02, x1 > -16.16}'},
{'diff': '{x2 <= -106.13, x2 > -215.27, x2 > -107.98, x2 <=
    -107.07}'},
{'diff': '{x2 > -106.13, x1 > -204.00, x1 <= 152.57, x1 <=
    -7.30, x2 <= 96.83, x1 > -9.21}'},
{'diff': '{x2 > -106.13, x1 > -204.00, x1 <= 152.57, x1 <=
    -7.30, x2 > 96.83, x1 <= -8.96}'},
{'diff': '{x2 > -106.13, x1 > -204.00, x1 <= 152.57, x1 > -7.30,
    x2 <= 96.85}'}
```

Thus, the differences are more difficult to explain and understand. That confirms that an optimized process must be found to generate local diff-datasets that contains increased instances close to all decision boundaries. Listing 7.4 shows the decision rules derived from the multiclass decision tree (see Figure 7.19b). Each set of rules represents a path of the decision tree from the root to a “01” or “10” diff-leaf. Again, we only show the “diff” decision rule paths. In the case of the multiclass setting, the differences are depicted compactly and comprehensibly by using the derived decision rules.

Listing 7.4: Decision rules - global multiclass diff-classifier

```
{'01': '{x1 <= -8.78, x2 <= 96.84, x1 > -15.05, x2 <=
    -111.70}'},
{'01': '{x1 <= -8.78, x2 > 96.84, x1 > -203.02}'},
{'10': '{x1 > -8.78, x1 <= 151.30, x2 > -108.31, x2 <= 96.93}'},
```

Comparison of the Data Approaches

A crucial advantage of the local genetic methods is that increased instances can be generated close to the to be recognized instances synthetically, especially around the closer decision boundaries. That means these approaches are not dependent on existing instances. In contrast, the real data approaches are depending on the existing data. In the case of the local approaches, we also show that the selection of different to be recognized instances can be used to generate various diff-datasets. The concatenation of the generated diff-datasets to one global dataset helps train a global diff-classifier that can recognize the global differences between the black boxes. In the case of the running example dataset, we can show that both genetic neighborhood approaches create increased instances around the existing boundaries of the difference-classes. That enables us to illustrate by using plots that the decision boundaries can be trained accurately, and the genetic approaches can recognize all global differences between the black boxes. Also, we can observe that for the running example, the initial genetic approach generates more instances around the boundaries than the modified approach. However, the modified approach specifically generates increased synthetic instances in the closest possible vicinity close to a particular instance to be recognized. Also, the performance measurement shows that the diff-classifier based on the modified approach performs more accurately to predict differences. That means, to recognize the global differences based on the modified approach, enough instances to generate various diff-datasets must be selected.

7.1.2 Diagonal Example Dataset

Now, we present the diagonal example dataset for illustrating how the decision tree-based diff-classifier of DiRo2C can handle a non-orthogonal decision boundary. For this, we use the two-dimensional synthetic classification dataset[2] as seen in Figure 7.20 to train the black boxes. This example can be compared with the mentioned *difference classification problem* diagonal example (see Figure 3.2). For that example, we initially create the right shown dataset for black box B. The dataset for black box B shows a two-dimensional dataset with the continuous features x_1 and x_2. It contains 300 instances (datapoints) with the following properties for feature x_1: $min = -126.19$, $max = 362.16$, $\mu = 139.14$, and $\sigma = 77.79$ and with the following properties for feature x_2: $min = -429.45$, $max = 265.10$, $\mu = 9.98$, and $\sigma = 157.80$. The instances of the datasets are classified into two classes “0” and “1”. 150 instances of the dataset of black box B are

[2]Datasets are available under: https://doi.org/10.5281/zenodo.5701440

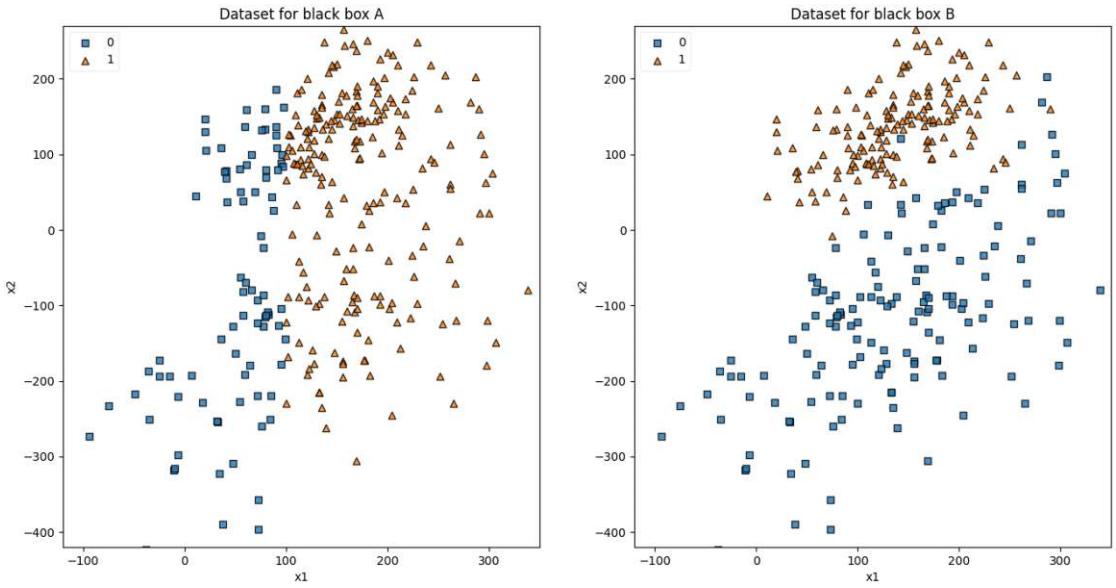


Figure 7.20: Diagonal example of a synthetic classification dataset for black box B and a manipulated dataset for black box A. The plot of the dataset of black box B shows the diagonal boundary between class "0" and "1".

assigned to the class "0", and 150 instances are assigned to the class "1". The instances of the dataset for black box B are generated by the sklearn "make_classification" function with the following parameters: make_classification(n_samples = 300, n_features = 2, n_informative = 2, n_clusters_per_class = 2, class_sep = 1.3, random_state = 0, flip_y = 0, scale = 100). The left dataset for black box A is manipulated on the basis of the dataset of black box B as follows: $(x_1 < 100 : y = 0)$, $(x_1 >= 100 : y = 1)$. 86 instances are now assigned to the class "0", and 214 instances are assigned to the class "1". Thus, we only manipulate the outcome classes of the instances of the dataset for black box B. The properties for the feature x_1 and x_2 remain unchanged.

Before we present how the decision tree-based diff-classifier handles a non-orthogonal decision boundary, we show the resulting decision boundaries (indicated by the black drawn lines) of the trained black box A and B in Figure 7.21. Both plots show how the instances are differently predicted by black box A and B. In our diagonal example, we split the generated dataset into a training and test dataset with a ratio of 80 to 20 percent. After that, we train black box A and B using the training dataset and a scikit-learn integrated Logistic Regression algorithm[3] with a defined random state of 0. The plot shows the resulting decision boundaries of black box A and B. The decision boundaries represent the learned regression function used for prediction. Black box A classifies the instances with the following trained decision boundary: $(x_1 < 100 : y = 0)$

[3]For further details see: `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`

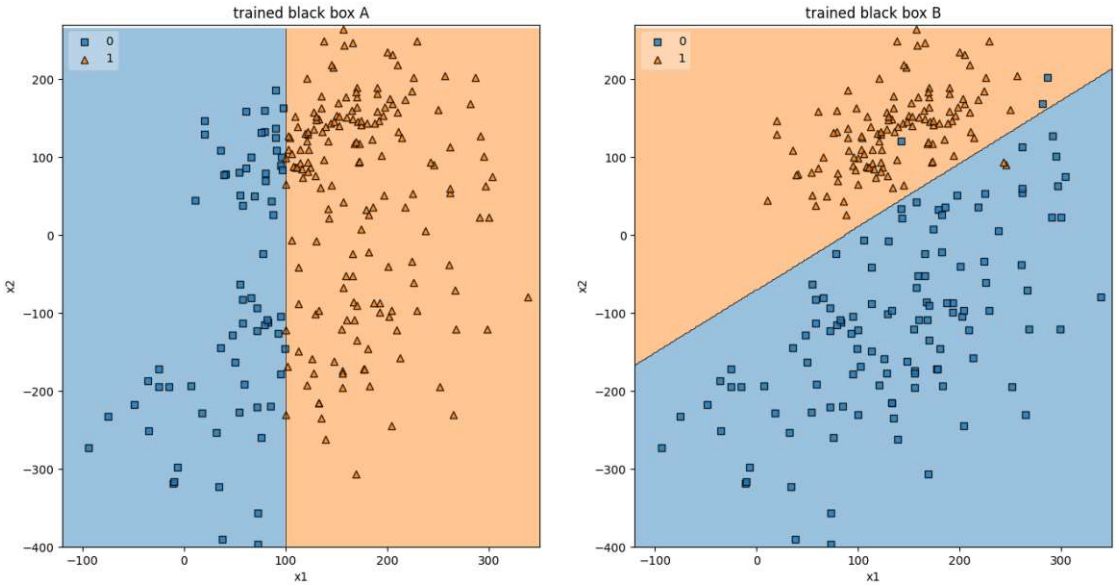


Figure 7.21: Trained black box classifier A and B with the corresponding decision boundaries based on the diagonal example dataset.

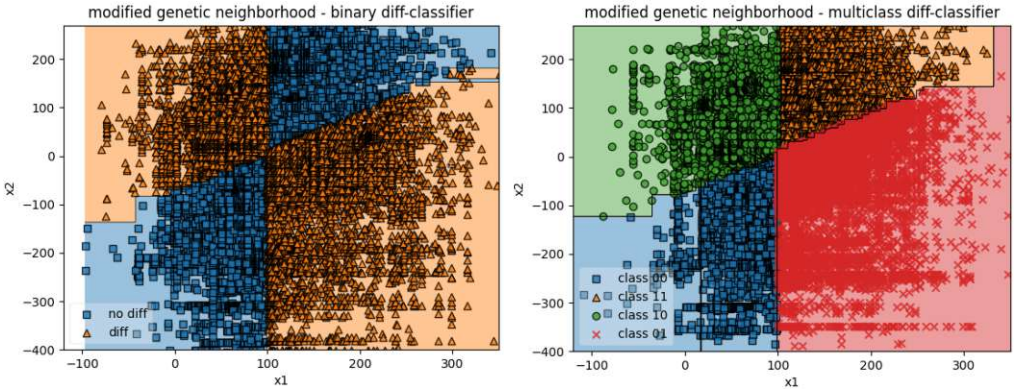
and ($x_1 >= 100 : y = 1$). The right plot shows the diagonal decision boundary of the trained black box classifier B.

Global Modified Genetic Neighborhood: Now, we present how the decision tree-based diff-classifier of DiRo2C can handle a non-orthogonal decision boundary. Therefore, we use a global diff-classifier trained by a global diff-dataset. Again, we concatenate the existing test datasets of black box A and B and select 10 instances for which the black boxes predict a different outcome and 10 instances for which the black boxes predict the same outcome. Next, we want to present the selected to be recognized instances for which we create single local diff-datasets and concatenate them to a global one:

- $X_{test}[2] : [150.53, 220.01]$
- $X_{test}[4] : [115.48, 186.17]$
- $X_{test}[7] : [186.16, 169.81]$
- $X_{test}[10] : [169.92, 165.67]$
- $X_{test}[12] : [-7.69, -221.87]$
- $X_{test}[13] : [48.94, -309.61]$
- $X_{test}[14] : [234.42, 82.90]$

- $X_{test}[15] : [21.77, 105.81]$
- $X_{test}[17] : [35.35, -429.45]$
- $X_{test}[18] : [92.86, 80.82]$
- $X_{test}[21] : [138.17, -89.86]$
- $X_{test}[22] : [123.77, -184.87]$
- $X_{test}[23] : [117.45, -56.10]$
- $X_{test}[24] : [168.26, 109.62]$
- $X_{test}[27] : [41.39, 68.95]$
- $X_{test}[32] : [60.69, -69.27]$
- $X_{test}[33] : [59.45, 137.44]$
- $X_{test}[35] : [128.21, 119.89]$
- $X_{test}[39] : [78.63, -86.81]$
- $X_{test}[42] : [209.71, 43.13]$

Figure 7.22 shows the generated global diff-datasets and the decision boundaries of the trained global diff-classifiers. The concatenated global binary diff-dataset contains 24,040 instances, whereby 9,672 instances are classified into class "no diff" and 14,368 instances are classified into class "diff". Now, we discuss the resulting decision boundaries of the binary diff-classifier. Figure 7.22a shows the generated global diff-dataset and the decision boundaries of the binary diff-classifier. The plot shows that not enough instances are generated in the outer right and left corners close to the decision boundary of black box B. Therefore, the global diff-classifier can't recognize in that area the diagonal boundary. Additionally, we show the data density of the generated global diff-dataset in Figure 7.23. For the x_1-axis, we use 60 bins, and for the x_2-axis, we also use 60 bins. The main problem we can observe is that the trained decision tree of the global diff-classifier is highly complex since the CART algorithm has to approximate the diagonal boundary, which leads to an increased generation of rules. Figure 7.24 shows the complex decision tree of the global binary diff-classifier. We can observe that the differences are difficult to explain and finally to understand. In Listing 7.5, we show the extracted decision rules of the binary diff-classifier. Again, we can observe that the diagonal boundary leads to an explanation that is hard to comprehend. Figure 7.22b shows the generated global diff-dataset and the decision boundaries of the multiclass diff-classifier. We can observe in the multiclass setting the same effect as in the binary setting. The diagonal decision boundary has to be approximated by the CART algorithm by creating increased rules. The concatenated global multiclass diff-dataset contains 21,904 instances, whereby 3,717



(a) Decision boundaries of binary diff-classifier

(b) Decision boundaries of multiclass diff-classifier

Figure 7.22: Global diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the diagonal example dataset.

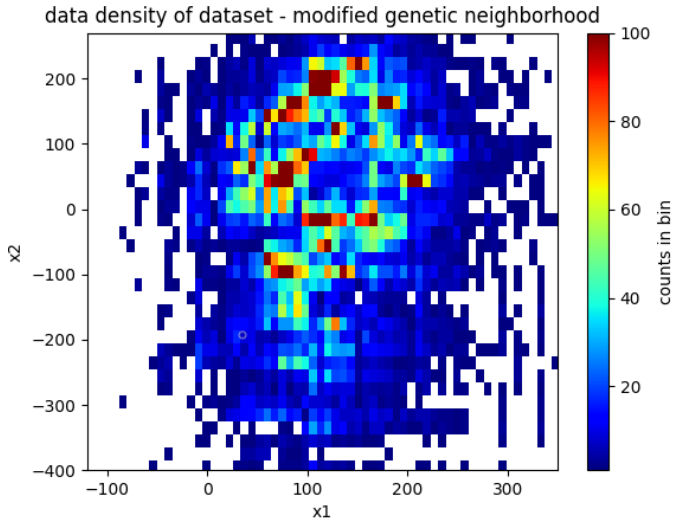


Figure 7.23: Data density of the global modified genetic neighborhood dataset based on the diagonal example. Figure 7.22b shows the generated global datasets using the modified approach for various located instances.

instances are classified into class “00”, 5,835 instances are classified into class “11”, 5,467 instances are classified into class “10”, and 6,885 instances are classified into class “01”. Figure 7.24 shows the decision tree of the global multiclass diff-classifier.
We will come back to the problem described above in Chapter 8.

Listing 7.5: Decision rules - global binary diff-classifier

```
{'diff': '{x2 <= 159.62, x2 <= -60.12, x1 <= 99.33, x1 <=
```

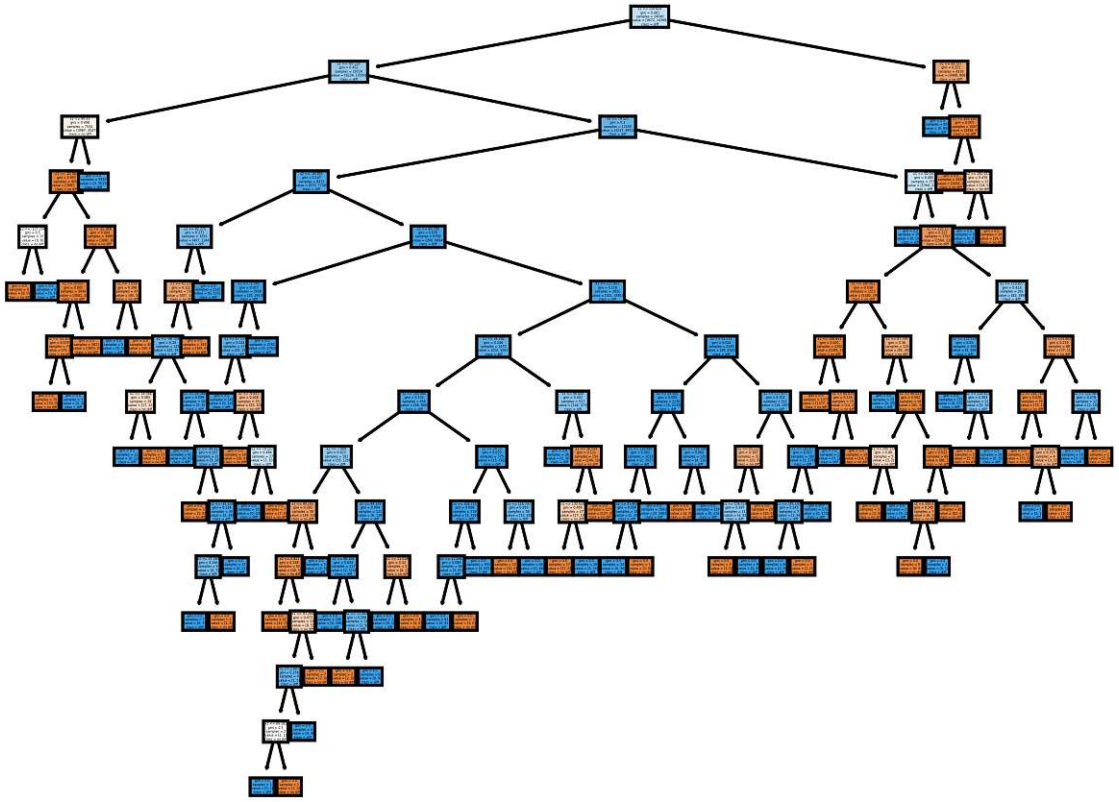


Figure 7.24: Decision tree of the global binary diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the diagonal example.

```
    −42.56, x2 > −137.26}'},
{'diff': '{x2 <= 159.62, x2 <= −60.12, x1 <= 99.33, x1 > −42.56,
    x2 <= −61.46, x1 <= −8.65, x2 > −82.82}'},
{'diff': '{x2 <= 159.62, x2 <= −60.12, x1 <= 99.33, x1 > −42.56,
    x2 > −61.46, x1 <= 7.02}'},
{'diff': '{x2 <= 159.62, x2 <= −60.12, x1 > 99.33}'},
{'diff': '{x2 <= 159.62, x2 > −60.12, x2 <= 79.52, x2 <= −26.95,
    x1 <= 99.37, x1 <= 45.10, x2 <= −48.70, x1 <= 18.36}'},
{'diff': '{x2 <= 159.62, x2 > −60.12, x2 <= 79.52, x2 <= −26.95,
    x1 <= 99.37, x1 <= 45.10, x2 > −48.70, x1 <= 29.38}'},
{'diff': '{x2 <= 159.62, x2 > −60.12, x2 <= 79.52, x2 <= −26.95,
    x1 <= 99.37, x1 <= 45.10, x2 > −48.70, x1 > 29.38, x2 >
    −44.81, x2 <= −35.13, x1 <= 37.58}'},
{'diff': '{x2 <= 159.62, x2 > −60.12, x2 <= 79.52, x2 <= −26.95,
    x1 <= 99.37, x1 <= 45.10, x2 > −48.70, x1 > 29.38, x2 >
    −44.81, x2 > −35.13}'},
{'diff': '{x2 <= 159.62, x2 > −60.12, x2 <= 79.52, x2 <= −26.95,
```

```
    x1 > 99.37}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 <= 86.79, x2 <= -5.79, x1 <= 67.15}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 <= 86.79, x2 <= -5.79, x1 > 67.15, x2 > -9.85, x1 <=
   78.10}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 <= 86.79, x2 > -5.79}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 <= 34.28, x1 <= 100.86, x2 <=
   7.88, x1 <= 99.20, x2 > 0.82, x1 <= 93.26, x2 <= 2.68, x1 <=
   89.18}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 <= 34.28, x1 <= 100.86, x2 <=
   7.88, x1 <= 99.20, x2 > 0.82, x1 <= 93.26, x2 > 2.68}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 <= 34.28, x1 <= 100.86, x2 <=
   7.88, x1 > 99.20}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 <= 34.28, x1 <= 100.86, x2 >
   7.88, x1 <= 99.10, x1 <= 98.35}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 <= 34.28, x1 <= 100.86, x2 >
   7.88, x1 <= 99.10, x1 > 98.35, x2 > 11.26}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 <= 34.28, x1 <= 100.86, x2 >
   7.88, x1 > 99.10, x2 <= 11.03}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 <= 34.28, x1 > 100.86, x2 <=
   24.14, x1 <= 102.71, x2 <= 11.26}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 <= 34.28, x1 > 100.86, x2 <=
   24.14, x1 > 102.71}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 <= 34.28, x1 > 100.86, x2 >
   24.14, x1 > 116.74}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 > 34.28, x1 <= 99.40}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 <= 137.80, x2 > 34.28, x1 > 99.40, x2 <= 38.54,
    x1 > 130.62}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
   x1 > 86.79, x1 > 137.80, x2 <= 61.23, x2 <= 53.56, x1 <=
```

```
    139.68, x2 <= 41.90}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
    x1 > 86.79, x1 > 137.80, x2 <= 61.23, x2 <= 53.56, x1 >
    139.68}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
    x1 > 86.79, x1 > 137.80, x2 <= 61.23, x2 > 53.56, x1 >
    158.58}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
    x1 > 86.79, x1 > 137.80, x2 > 61.23, x1 <= 177.67, x2 <=
    65.12, x1 > 158.51}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
    x1 > 86.79, x1 > 137.80, x2 > 61.23, x1 > 177.67, x1 <=
    183.67, x1 <= 181.14}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 <= 79.52, x2 > -26.95,
    x1 > 86.79, x1 > 137.80, x2 > 61.23, x1 > 177.67, x1 >
    183.67}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 > 79.52, x1 <=
    99.06}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 > 79.52, x1 > 99.06, x1
    <= 220.12, x1 <= 190.90, x1 > 188.83, x2 <= 84.87}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 > 79.52, x1 > 99.06, x1
    <= 220.12, x1 > 190.90, x2 <= 87.76}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 > 79.52, x1 > 99.06, x1
    <= 220.12, x1 > 190.90, x2 > 87.76, x2 <= 96.02, x1 >
    200.17}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 > 79.52, x1 > 99.06, x1
    <= 220.12, x1 > 190.90, x2 > 87.76, x2 > 96.02, x2 <= 100.80,
     x2 > 100.60}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 > 79.52, x1 > 99.06, x1
    > 220.12, x2 <= 119.82, x2 <= 110.52}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 > 79.52, x1 > 99.06, x1
    > 220.12, x2 <= 119.82, x2 > 110.52, x1 > 228.61}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 > 79.52, x1 > 99.06, x1
    > 220.12, x2 > 119.82, x1 <= 254.48, x1 > 244.76, x2 <=
    130.37}'},
{'diff': '{x2 <= 159.62, x2 > -60.12, x2 > 79.52, x1 > 99.06, x1
    > 220.12, x2 > 119.82, x1 > 254.48, x2 <= 152.98}'},
{'diff': '{x2 > 159.62, x1 <= 99.15}'},
{'diff': '{x2 > 159.62, x1 > 99.15, x1 > 295.28, x2 <= 182.05}'}
```

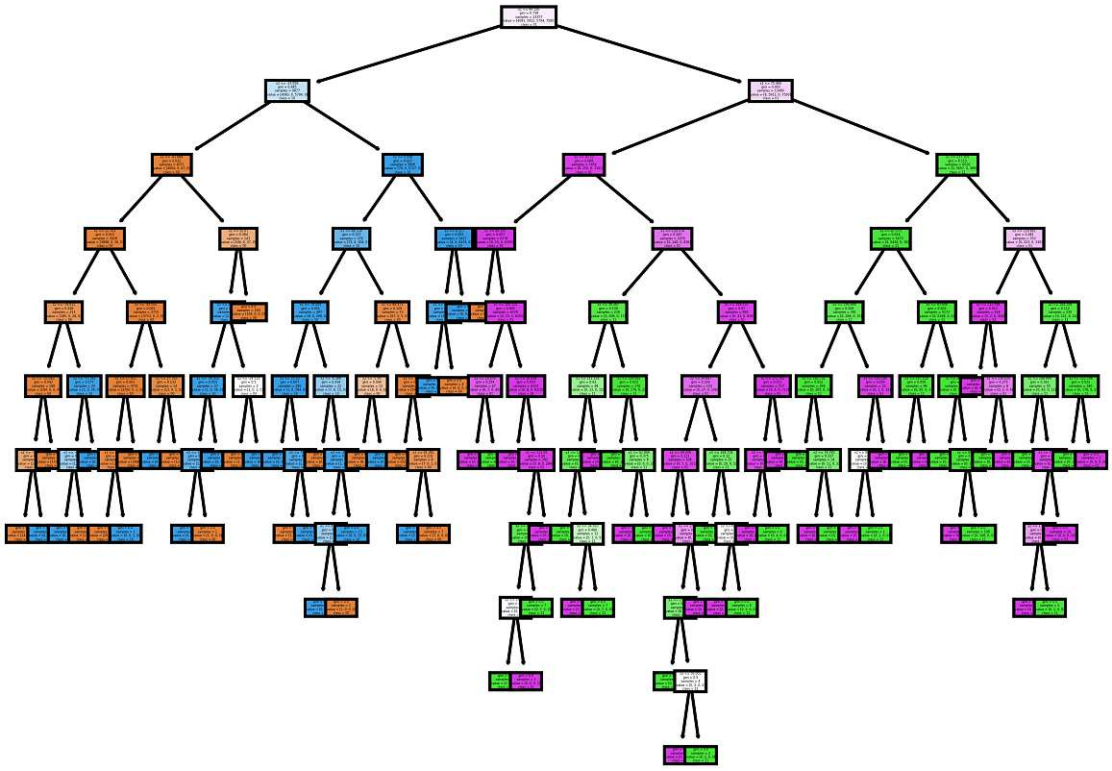


Figure 7.25: Decision tree of the global multiclass diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the diagonal example.

7.2 Adult Dataset

This section presents the results of the *adult* dataset. We have already presented the *adult* dataset in detail in Chapter 5. We also use the same pre-processing and black box training steps as explained in the mentioned chapter. The main difference is that we carry out two different manipulation scenarios for the following evaluation. Those scenarios are explained in detail below. Thus, again we use for training black boxes A and B the decision tree-based CART algorithm. For the evaluation, we select for the local data approaches various located instances to be recognized, generate for each of them a particular diff-dataset, and concatenate the resulting diff-dataset to a global one. Finally, we train a global diff-classifier based on the global diff-dataset. For experimental purposes, we select 10 instances from the existing (entire) dataset for which the black boxes predict a different outcome and 10 instances for which the black boxes predict the same outcome. We use the entire dataset for that experimental setting since otherwise, we cannot select enough to be recognized instances to show the global effect of the local data approaches. We present the selected instances for each manipulation scenario below. The decision tree classifiers for black boxes A and B are trained with a defined maximum tree depth to evaluate the detected differences more compactly and clearly. However, additionally,

we present an example of classifiers for black box A and B that are unlimited concerning the maximum tree depth. We show the different decision trees of the black boxes for each scenario. For the first scenario, we show the decision trees per data approach. For the other scenarios, we only show the results of the global diff-classifier based on the modified genetic neighborhood approach. Now, we present our results using the *adult* dataset. To show that the diff-classifier truly recognizes the differences, we use different manipulation scenarios. We describe each manipulation scenario in detail and present the results afterward. For each manipulation scenario, all instances for the dataset of black box B are manipulated by the same factor. Each case contains a hypothesis of how the manipulation should affect the decision tree of the diff-classifier.

7.2.1 Manipulation of an Influential Feature (DMS1)

To illustrate the effect for that scenario, we manipulate an influential feature by changing the value for all instances by a certain factor. That manipulation should be reflected and thus be recognizable and displayed in the decision tree of the diff-classifier. For identifying an influential feature, we analyze the initial unchanged decision tree of black box A (see Figure 7.26) trained with a maximum tree depth of 6. We choose the Feature "hours-per-week" and manipulate it as follows: $x_{hours-per-week} = x_{hours-per-week} + 10 \; \forall x \in X$. That means we add the value 10 to the Feature "hours-per-week" for each instance x of X. Figure 7.27 shows the decision tree of the manipulated black box B. The yellow marked node shows the changed rule for the Feature "hours-per-week" of the manipulated black box B. The condition is now shifted by plus 10.

We select the following instances to train the global diff-classifiers for the local data approaches:

- Features of X: ['age', 'workclass', 'education', 'marital-status', 'occupation', 'relationship', 'race', 'sex', 'capital-gain', 'capital-loss', 'hours-per-week', 'native-country']

Instances where the black boxes predict the same outcome:

- $X[1] : [39, 6, 9, 4, 0, 1, 4, 1, 2174, 0, 40, 38]$
- $X[2] : [50, 5, 9, 2, 3, 0, 4, 1, 0, 0, 13, 38]$
- $X[3] : [38, 3, 11, 0, 5, 1, 4, 1, 0, 0, 40, 38]$
- $X[4] : [53, 3, 1, 2, 5, 0, 2, 1, 0, 0, 40, 38]$
- $X[5] : [28, 3, 9, 2, 9, 5, 2, 0, 0, 0, 40, 4]$
- $X[6] : [37, 3, 12, 2, 3, 5, 4, 0, 0, 0, 40, 38]$
- $X[7] : [49, 3, 6, 3, 7, 1, 2, 0, 0, 0, 16, 22]$

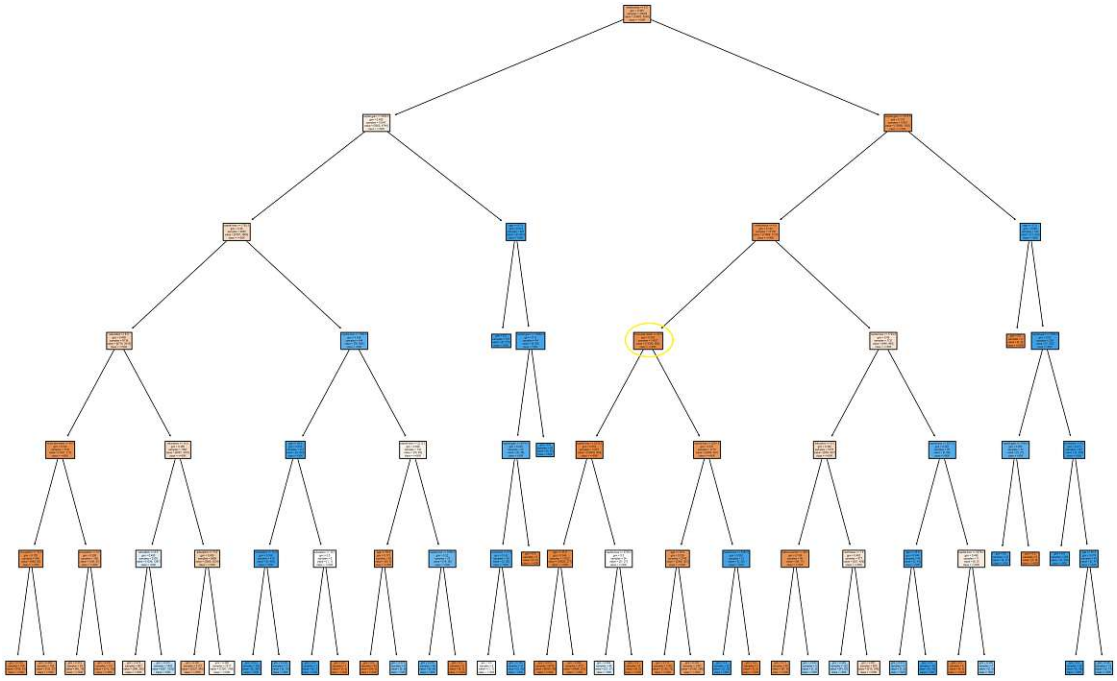


Figure 7.26: Decision tree of black box A using the adult dataset for DMS1 with a maximum tree depth of 6.

- $X[8] : [52, 5, 11, 2, 3, 0, 4, 1, 0, 0, 45, 38]$
- $X[9] : [31, 3, 12, 4, 9, 1, 4, 0, 14084, 0, 50, 38]$
- $X[10] : [42, 3, 9, 2, 3, 0, 4, 1, 5178, 0, 40, 38]$

Instances where the black boxes predict a different outcome:

- $X[382] : [46, 3, 4, 0, 2, 1, 4, 0, 0, 2339, 45, 38]$
- $X[6126] : [40, 3, 9, 4, 12, 1, 1, 0, 0, 2258, 48, 29]$
- $X[10503] : [29, 3, 9, 4, 3, 1, 4, 0, 0, 2258, 45, 38]$
- $X[13329] : [49, 4, 15, 0, 3, 1, 4, 1, 0, 2339, 50, 38]$
- $X[15037] : [55, 3, 15, 4, 0, 1, 2, 0, 0, 2339, 45, 38]$
- $X[17628] : [26, 3, 9, 4, 3, 1, 4, 1, 0, 2258, 45, 38]$
- $X[19536] : [27, 5, 11, 4, 2, 1, 4, 1, 0, 2258, 50, 38]$
- $X[20970] : [34, 3, 12, 4, 9, 1, 4, 1, 0, 2258, 50, 38]$

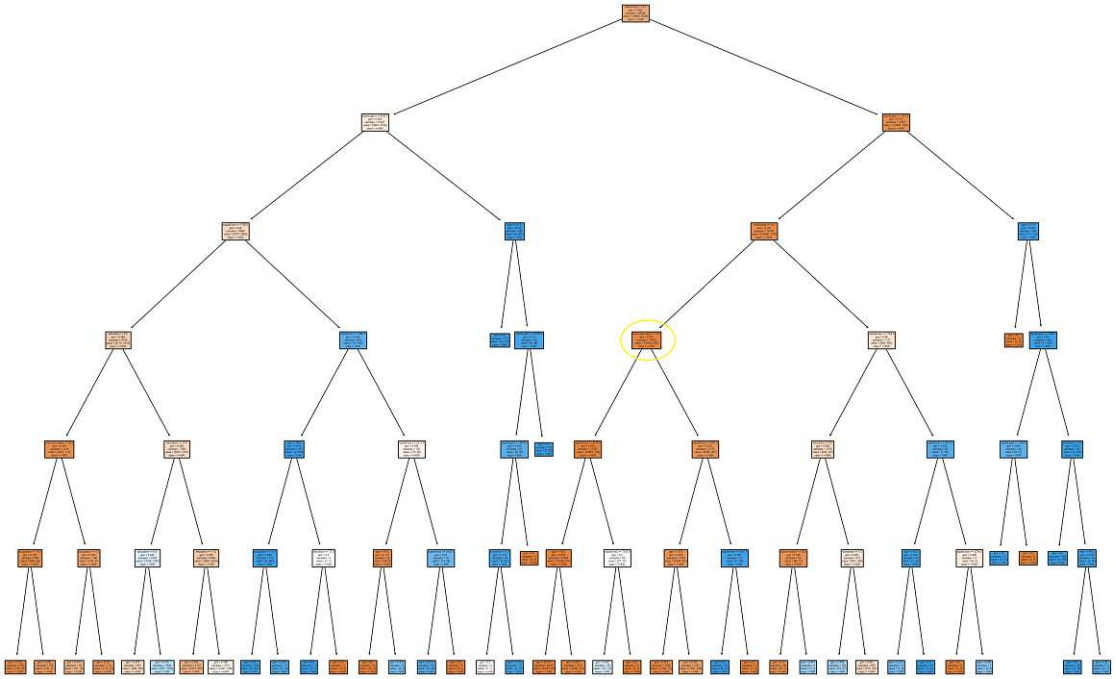


Figure 7.27: Decision tree of black box B using the adult dataset for DMS1 with a maximum tree depth of 6.

- $X[28174] : [49, 6, 10, 4, 3, 1, 4, 0, 0, 2258, 50, 38]$
- $X[31959] : [45, 3, 11, 0, 4, 1, 4, 1, 0, 2258, 44, 38]$

Global Modified Genetic Neighborhood Data

Now, we present the recognized differences of the diff-classifiers for DMS1 based on the modified genetic neighborhood data approach. At first, we discuss the results of the global binary diff-classifier. The concatenated global diff-dataset contains 21,857 instances, whereby 17,186 instances are classified into class "no diff" and 4,671 instances are classified into class "diff". Since we perform an instance unique-check globally, the result is an unbalanced global diff-dataset. Figure 7.28 shows the decision tree of the global binary diff-classifier. The tree contains the following two nodes with the rules in the upper area of the tree: The rule $if\ hours - per - week > 43.5$ (right branch) and the following rule $if\ hours - per - week <= 53.5$ (left branch) that lead to a "diff" leaf. Those two rules are included in the set of rules that explain the recognized differences between black box A and B (see Listing 7.6). Figure 7.26 shows the decision tree of black box A, and it contains the following node with the rule: $if\ hours - per - week <= 43.5$. For comparison, Figure 7.27 shows the decision tree of the manipulated black box B, and it contains the following node with the shifted rule: $if\ hours - per - week <= 53.5$. The decision tree of the diff-classifier shows that the manipulated differences between black

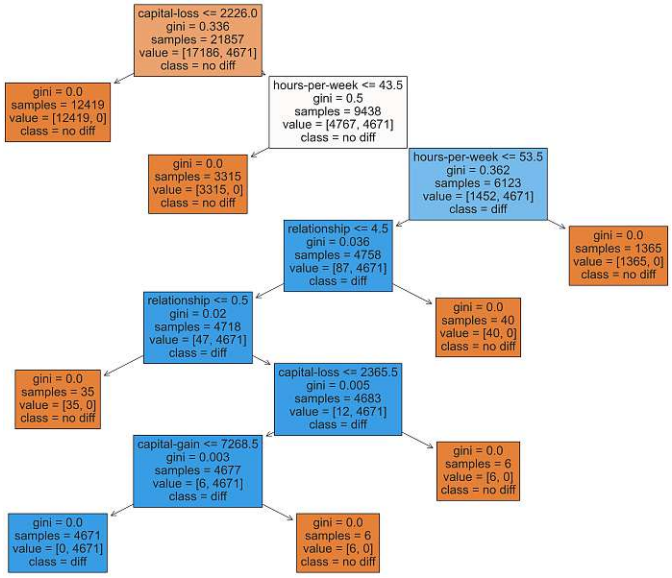


Figure 7.28: Decision tree of the global binary diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the adult dataset for DMS1.

Listing 7.6: Decision rules - global binary diff-classifier

```
{'diff': '{capital-loss > 2226.0, hours-per-week > 43.5, hours-
    per-week <= 53.5, relationship <= 4.5, relationship > 0.5,
    capital-loss <= 2365.5, capital-gain <= 7268.5}'}
```

box A and B can be recognized (see Figure 7.28). However, the diff-classifier also contains various nodes and conditions for features that are not affected by the manipulation.

Now, we show the resulting decision tree of the global multiclass diff-classifier. The multiclass diff-dataset contains 22,471 instances, whereby 11,469 instances are classified into class “00”, 6,256 instances are classified into class “11”, 4,746 instances are classified into class “10”, and 0 instances are classified into class “01”. Figure 7.29 shows the decision tree of the multiclass diff-classifier. Again, the tree contains the following two nodes with the rules in the upper area of the tree: The rule $if\ hours-per-week > 43.5$ (right branch) and the following rule $if\ hours-per-week <= 53.5$ (left branch) that lead to a “10” leaf. Compared to the decision tree of the binary diff-classifier, there are much more recognized non-diff (difference-class “00” and “11”) paths. Apart from the poorer performance of the multiclass diff-classifier, the advantage of the multiclass approach is that the exactly predicted results of black box A and B can be determined.

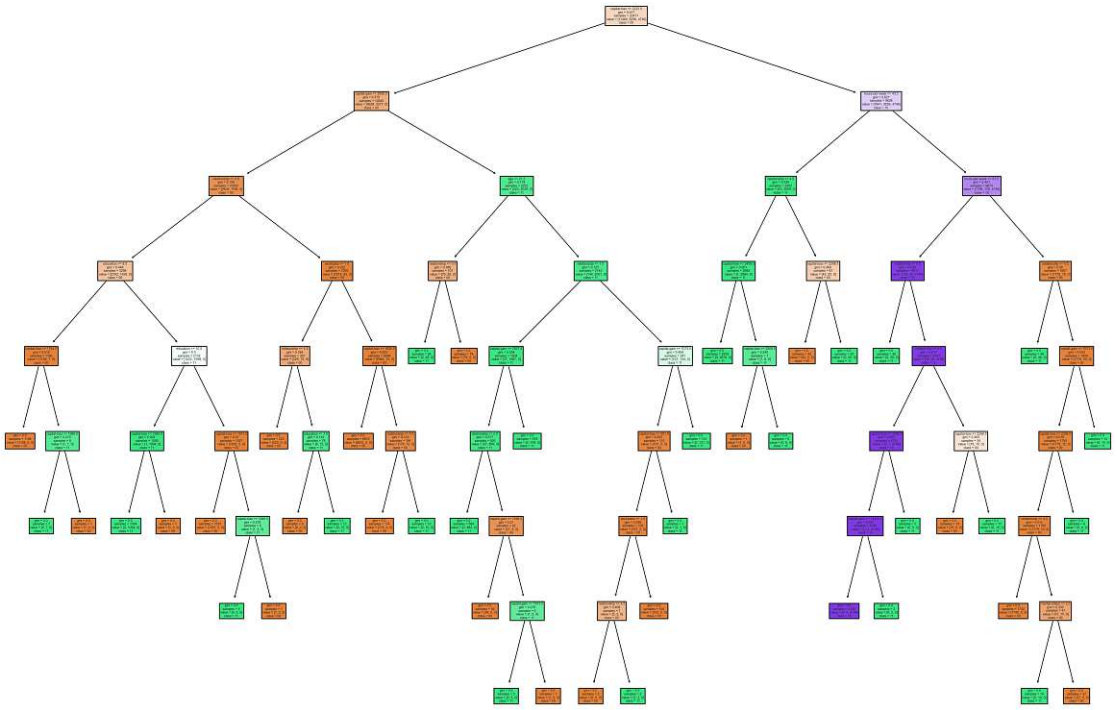


Figure 7.29: Decision tree of the global multiclass diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the adult dataset for DMS1

Listing 7.7: Decision rules - global multiclass diff-classifier

```
{'10': '{capital-loss > 2252.0, hours-per-week > 43.5, hours-
    per-week <= 53.5, relationship > 0.5, relationship <= 4.5,
    capital-loss <= 2365.5, capital-gain <= 13450.0}'}
```

Unlimited Maximum Depth Tree

We also want to show an example where black box A and B are trained without limitation concerning the depth tree. In that case, we manipulate again the Feature "hours-per-week" and manipulate it again as follows:
$x_{hours-per-week} = x_{hours-per-week} + 10 \; \forall x \in X$. That means we add the value 10 to the Feature "hours-per-week" for each instance x of X. Figure 7.30 shows the decision tree of black box A and Figure 7.31 shows the decision tree of black box B. Now, the decision trees contain many nodes depending on the Feature "hours-per-week". In principle, all conditions depending on the Feature "hours-per-week" of the decision tree for black box B are shifted by plus 10.

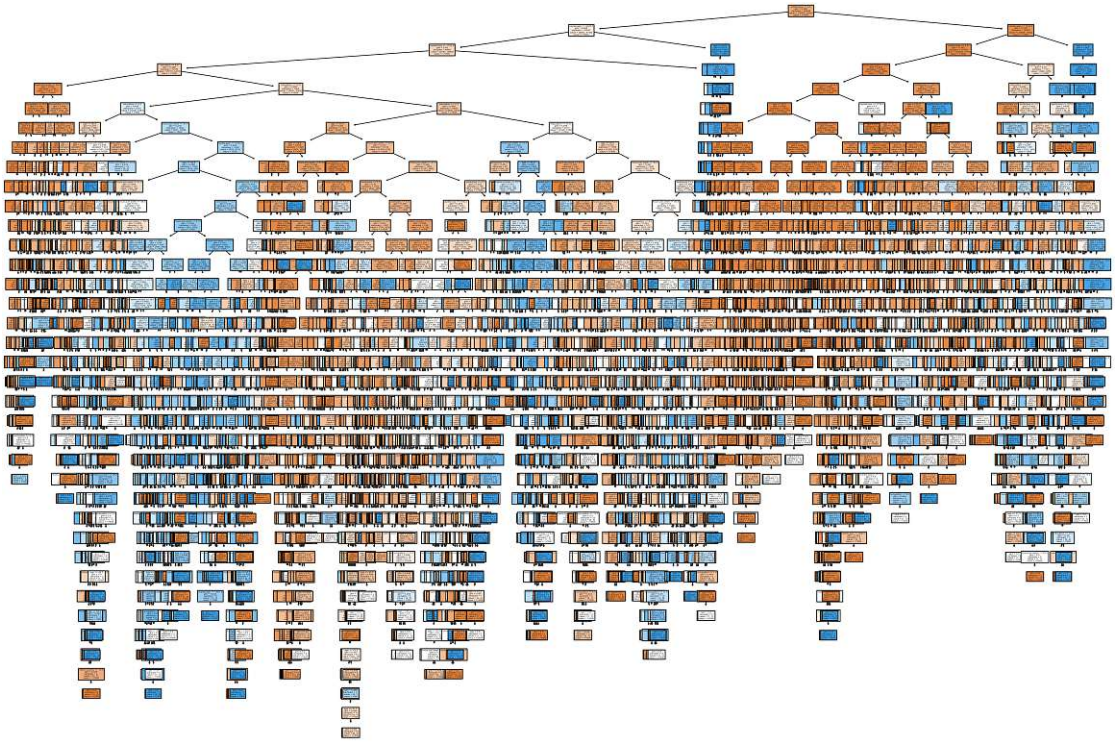


Figure 7.30: Unlimited decision tree of black box A using the adult dataset for DMS1 with no defined maximum tree depth.

Global Modified Genetic Neighborhood Data: Now, we present in Figure 7.32 the recognized differences of the global binary diff-classifier for DMS1. We generate the single diff-datasets again with the mentioned above instances to recognize. Then, we concatenate the diff-dataset to a global one and train the global binary diff-classifier. With this example, we want to show how complex the decision tree can become even though a forced targeted manipulation of a specific influential feature is performed. The decision tree shows that the recognized differences are no longer easy to comprehend for a human. We only present the results of the global binary diff-classifier since the decision tree of the global multiclass diff-classifier is also not comprehensible.

7.2.2 Manipulation of an Irrelevant Feature (DMS2)

In the next scenario, we manipulate the dataset for black box B as follows. Across all data points, we manipulate an irrelevant feature by changing the value by a specific factor. Therefore, those particular manipulations shouldn't be reflected and thus not be recognizable and displayed in the decision tree of the diff-classifier. we analyze again the initial unchanged decision tree of black box A for DMS1 (see Figure 7.26) trained with a maximum tree depth of 6 to identify an irrelevant feature. We choose the Feature "marital-status" and manipulate it as follows: $x_{marital-status} = (x_{marital-status} + 1) \mod 7 \, \forall x \in X$.

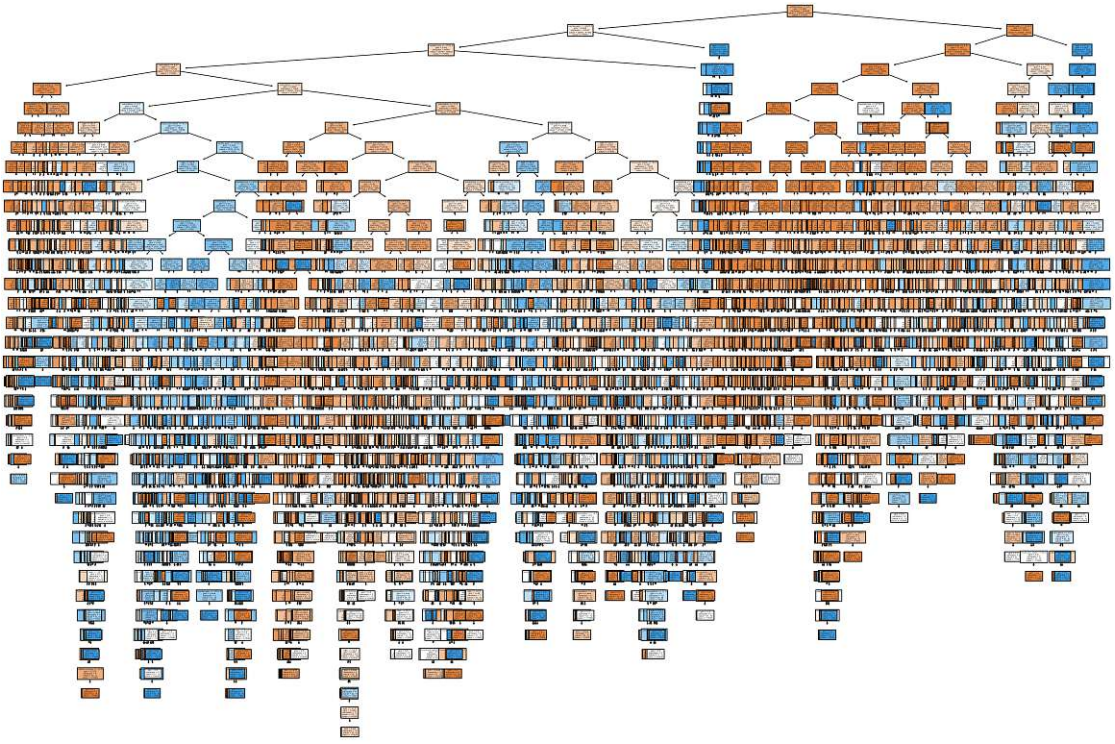


Figure 7.31: Unlimited decision tree of black box B using the adult dataset for DMS1 with no defined maximum tree depth.

The discrete Feature “marital-status” contains 7 different values, which are encoded as “0”, “1”, ..., and “6”. To shift the value of the Feature “marital-status” by one for each x, $x_{marital-status}$ plus 1, and then modulo the number of the possible values of $x_{marital-status}$ is performed. Therefore, we shift the label encoded values plus 1 except the last encoded value, which is shifted from “6” to “0”. Figure 7.33 shows the decision tree of the manipulated black box B. Now, no differences exist between the initial decision tree of black box A (see Figure 7.26) and black box B since the Feature “marital-status” is irrelevant for the classification.

We select the following instances to train the global diff-classifiers for the local data approaches:

- Features of X: [‘age’, ‘workclass’, ‘education’, ‘marital-status’, ‘occupation’, ‘relationship’, ‘race’, ‘sex’, ‘capital-gain’, ‘capital-loss’, ‘hours-per-week’, ‘native-country’]

Instances where the black boxes predict the same outcome:

- $X[1] : [39, 6, 9, 4, 0, 1, 4, 1, 2174, 0, 40, 38]$

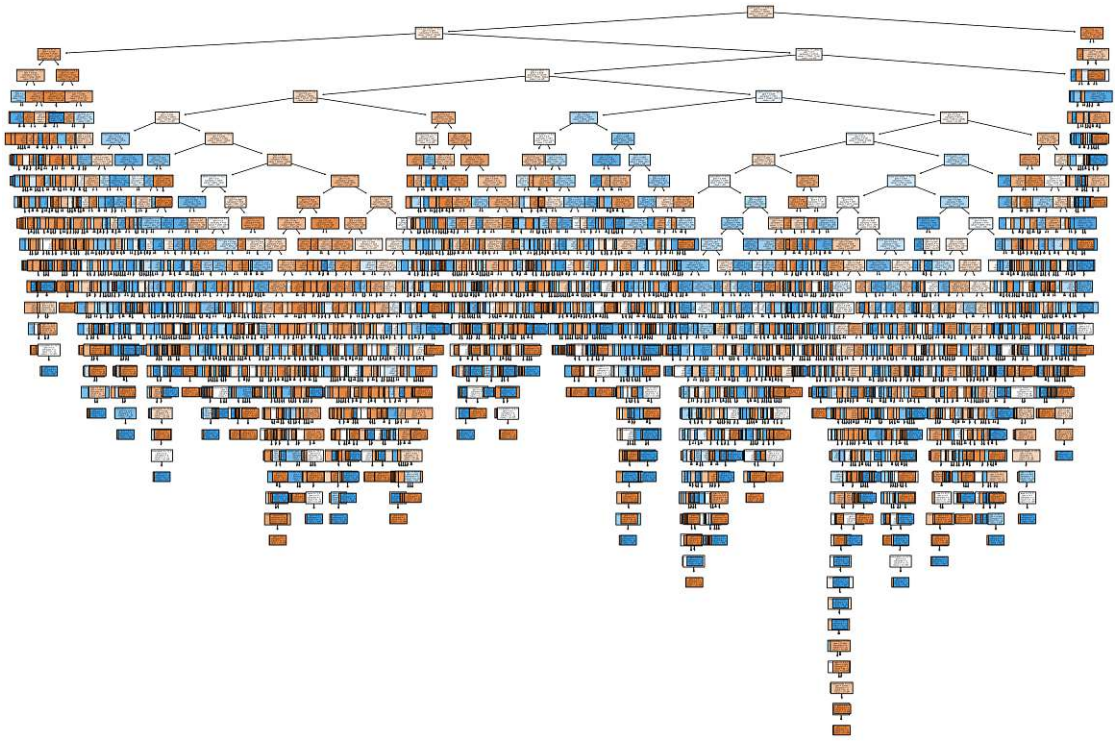


Figure 7.32: Decision tree of the global binary diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the unlimited adult dataset for DMS1.

- $X[2] : [50, 5, 9, 2, 3, 0, 4, 1, 0, 0, 13, 38]$
- $X[3] : [38, 3, 11, 0, 5, 1, 4, 1, 0, 0, 40, 38]$
- $X[4] : [53, 3, 1, 2, 5, 0, 2, 1, 0, 0, 40, 38]$
- $X[5] : [28, 3, 9, 2, 9, 5, 2, 0, 0, 0, 40, 4]$
- $X[6] : [37, 3, 12, 2, 3, 5, 4, 0, 0, 0, 40, 38]$
- $X[7] : [49, 3, 6, 3, 7, 1, 2, 0, 0, 0, 16, 22]$
- $X[8] : [52, 5, 11, 2, 3, 0, 4, 1, 0, 0, 45, 38]$
- $X[9] : [31, 3, 12, 4, 9, 1, 4, 0, 14084, 0, 50, 38]$
- $X[10] : [42, 3, 9, 2, 3, 0, 4, 1, 5178, 0, 40, 38]$

Since there are no differences between black box A and B, we cannot select any instances where the black boxes predict a different outcome.

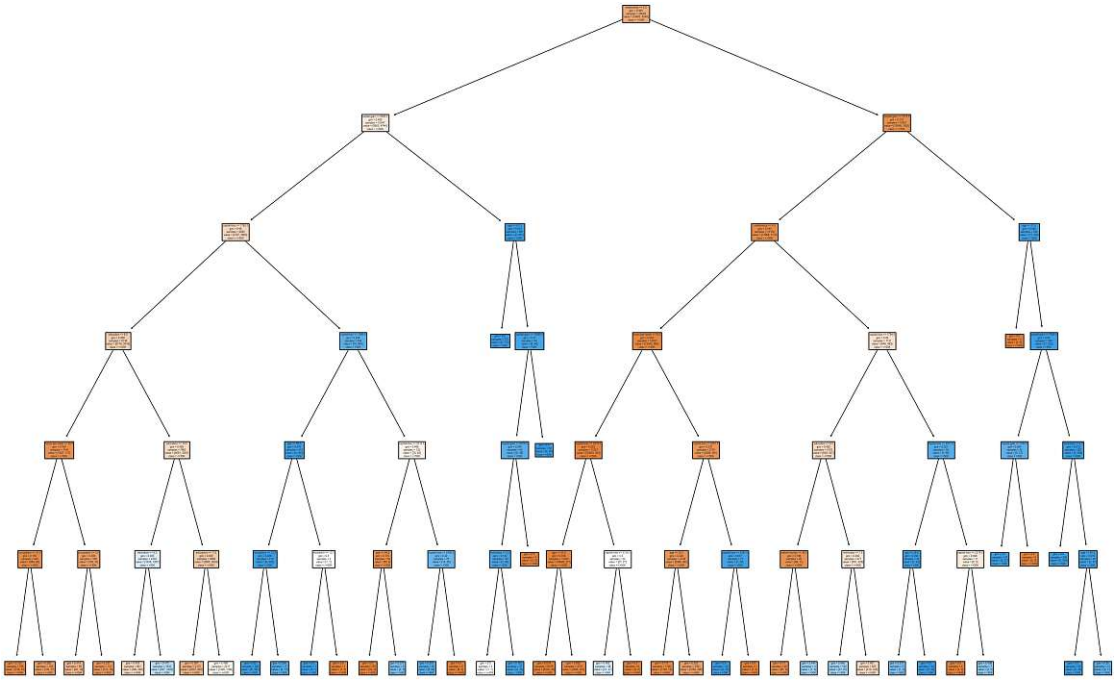


Figure 7.33: Decision tree of black box B using the adult dataset for DMS2 with a maximum tree depth of 6.

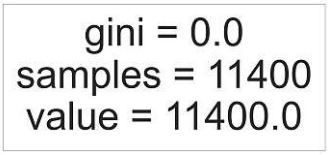


Figure 7.34: Decision tree of the global binary diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the adult dataset for DMS2.

Global Modified Genetic Neighborhood Data

Now, we present the recognized differences of the diff-classifiers for DMS2. At first, we discuss the results of the global binary diff-classifier. The concatenated global diff-dataset contains 11,400 instances, whereby all instances are classified into class “no diff” and 0 instances are classified into class “diff”. Figure 7.34 shows the decision tree of the global binary diff-classifier. The trained diff-classifier recognizes no differences as expected.

Next, we show the resulting decision tree of the global multiclass diff-classifier. The multiclass diff-dataset contains 11,742 instances, whereby 8,511 instances are classified into class “00”, 3,231 instances are classified into class “11”, 0 instances are classified into class “10”, and 0 instances are classified into class “01”. Figure 7.35 shows the decision tree of the multiclass diff-classifier. Again, as expected, the trained diff-classifier

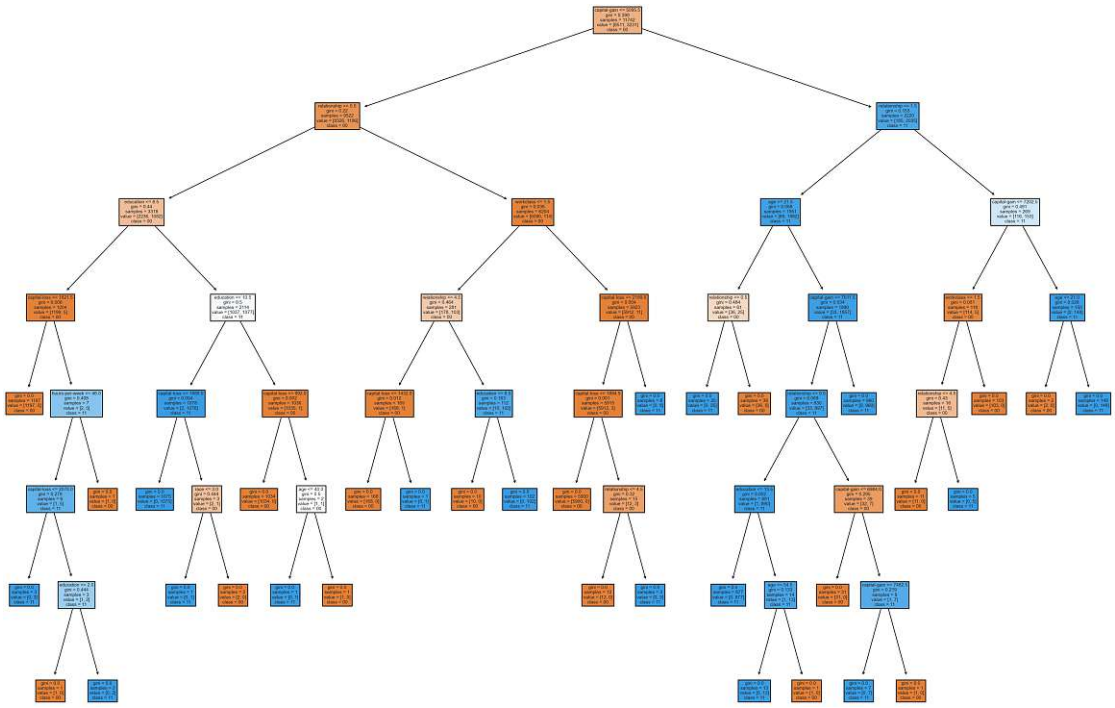


Figure 7.35: Decision tree of the global multiclass diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the adult dataset for DMS2

recognizes no differences. Although the decision tree contains more nodes than the binary decision tree, all paths lead to a non-difference outcome class.

7.3 Summary and Findings

We have evaluated the data approaches using a synthetic two-dimensional dataset. We could show that a global diff-classifier trained based on the modified genetic neighborhood approach can adequately recognize the (global) manipulated differences between black boxes A and B. However, we could observe that the initial genetic neighborhood approach can generate more instances in the relevant areas than the modified approach by evaluating the two-dimensional running example setting. The multiclass diff-classifier has the decisive advantage that every single possible difference-class combination can be determined. However, the modified genetic neighborhood data generation approach is a local data generation method. Therefore, the approach is dependent on a particular to be recognized instance x. That means instances are generated locally as close as possible compared to x. Hence, we also show the impact of generating for different located instances, local neighborhood diff-datasets, and concatenate them to a global one. We also evaluate our approach using the well-known benchmark adult dataset. Therefore, we train a

global explainer using the global approach of generating diff-datasets for different located instances. We could observe that, in our described settings, this approach can explain the global differences between two black boxes adequately. Thus, we could confirm the stated hypothesis of the data manipulation scenarios. However, we also observe that the decision tree-based diff-classifier of DiRo2C can hardly explain differences when the difference decision boundary is non-orthogonal. We will take up this problem in the next chapter and suggest a potential solution.

CHAPTER 8

Conclusion and Future Work

In that last chapter, we summarize the most important findings of this work. In addition, we discuss the contributions compared to the state of the art. Finally, we give an outlook on future work and suggest potential solutions to known problems.

8.1 Conclusion

This master thesis has proposed an approach to provide an interpretable decision tree-based diff-classifier that explains the differences between two binary black box classifiers. That diff-classifier is interpretable by any model-agnostic method like LORE or LIME. DiRo2C trains that classifier by generating a diff-dataset Z_{Diff}. We present a modified genetic neighborhood data generation approach to provide multiple diff-dataset depending on a instance x to be recognized. That data generation approach performs best for training the predictor to recognize the difference compared to other data approaches. The modified genetic neighborhood approach provides a diff-dataset to train a local interpretable classification model. That means it is generating the data points depending on an instance x. By applying a genetic algorithm, it computes and creates artificially instances that are similar to the feature (value) characteristics of x. The modified approach produces a balanced dataset where roughly 50 percent instances show different results and 50 percent instances where the black boxes predict the same target class. Thus, it is (before the unique-check is performed) guaranteed that the data approach generates a balanced diff-dataset, and the trained diff-classifier recognizes the differences close to x. DiRo2C supports training a binary or a multiclass diff-classifier. The binary model predicts 0 for recognizing no difference and 1 for a difference. The multiclass predictor returns each possible decision combination of the two binary black box classifiers. For our evaluation, we measured the *Accuracy*, F_1-*score*, and *PearsonCC* metric for comparing the performance. According to the RQ, we show that the binary diff-classifier trained by the modified genetic neighborhood approach outperforms every other proposed method.

We also present how accurately the binary diff-classifier recognizes (depending on instance x) the local differences using manipulated datasets. The dataset was manipulated by applying different data manipulation scenarios to evaluate if the forced differences can be found in the resulting decision tree of the diff-classifier.

Contribution compared to the State of the Art

Therefore, compared to the State of the Art, we find a way to recognize accurately local differences between two black boxes. Hence, we modify the genetic neighborhood approach of LORE and change the fitness functions that the instances are created dependent on both black boxes. Furthermore, the local modified genetic neighborhood approach initially generates a balanced dataset Z_{Diff}. With this generated diff-dataset, we can train an accurate binary diff-classifier that predicts the local differences between the black boxes for a particular instance x. It is also possible to provide a multiclass diff-classifier that predicts every possible combination of the results of the black boxes. We also have shown that the local modified approach can be used to generate various diff-datasets for different located instances. That local diff-datasets can be concatenated to a global diff-dataset to train a global explainer for the differences between the black boxes. Since we provide a decision tree-based diff-classifier, we also enable the interpretation of the recognized differences by any model-agnostic method. Additionally, since we use the CART algorithm for our diff-classifier model, the predictor is up to a certain complexity inherently interpretable by analyzing the decision tree.

8.2 Future Work

This section is intended to provide an outlook for further work. First of all, we discuss the evaluated modified genetic neighborhood data generation approach of DiRo2C to name further improvements or extensions. We also discuss the "from local to global explanation" problem and the needed optimization concerning the local modified genetic neighborhood approach. Furthermore, we discuss further work in terms of combining of DiRo2C with model-agnostic methods, decision tree rule extraction, concept (data) drift detection, and recognition of differences between multiple binary black boxes.

Modified Genetic Neighborhood

The diff-classifier based on the local modified genetic neighborhood data generation approach can recognize local differences around a given instance x. Therefore, an optimization could be implemented in the future that guarantees that a balanced diff-dataset with no redundant instances is generated. However, the local modified genetic neighborhood is designed for generating a diff-dataset to recognize the local differences close to an instance x to recognized. But the modified approach can also be used to detect all the differences between the black boxes globally: As already mentioned in Section 5.1

the existing dataset is used to select the instances for which the differences are to be recognized. Therefore, a subsample of the dataset that covers the entire feature value space can be determined. But it would also be possible to generate generic instances for that black box A and B predict different results. That would guarantee independence from an existing data set. For each of those selected instances, a diff-dataset can be generated and combined to a global diff-dataset. Finally, a global explainer can be trained to recognize the overall differences between black box A and B. This approach can represent a potential solution to the generic problem of global explainable artificial intelligence by local explainers. Additionally, to generate for each instance of the subsample a diff-dataset Z_{Diff}, the method of data generation could be parallelized to reduce the run time.

Generalization of the Difference Classification Problem

DiRo2C can currently only detect the differences between two binary black box classifier. It could be extended, so that the difference between black box classifiers with *cn* different classes can be detected.

Recognition of Non-Orthogonal Decision Boundaries

DiRo2C uses for the recognition of the differences a particular Decision Tree Classification algorithm called CART. The problem is that such algorithm have to approximate a non-orthogonal decision boundary. Therefore, it must train many rules to be able to map the boundaries. Other interpretable ML models can be tested that map appropriately non-orthogonal decision boundaries. Oblique Decision Trees could offer a potential solution.

Combining of DiRo2C with Model-Agnostic Methods

DiRo2C provides a diff-classifier that enables the interpretation by any model-agnostic method like LORE or LIME. Providing a framework where the resulting diff-classifier of DiRo2C is combined with different model-agnostic methods would make it possible to interpret the diff-classifier using different explanatory approaches.

Rule Extraction from Decision Tree

Since the DiRo2C method trains a decision tree-based diff-classifier, the model is inherently interpretable. As already mentioned in Section 2.2, decision rules are another approach to explain why a predictor decided for a specific result. Therefore, the decision rules can be derived from the decision tree of the diff-classifier. The resulting rules can then be provided for better explainability (comprehensibility).

Concept (Data) Drift Detection

Regarding concept (data) drift detection, when deploying ML tools, it turns out that the input data could change over time. DiRo2C could be integrated into a concept (data) drift detection environment. If drifts are detected, a new classifier must be trained to solve the problem. After that, the differences between the old and the new classifier can be recognized by DiRo2C. This way you can understand the differences between the classifiers and decide whether you want to deploy and release the newly trained classifier.

Recognition of Differences between Multiple Binary Black Boxes

Currently it is only possible to recognize the differences between two binary black boxes with the help of DiRo2C. A difference detection of various versions of a classifier would also make sense with regard to concept (data) drift detection. For example, one approach could be to display the changes similar to a version history control. That means the changes for an instance x are made available in chronological order. In this context, DiRo2C can be combined with a model-agnostic method to display appropriate explanations.

List of Figures

1.1 Problem statement illustration 4
1.2 Adapted design science research cycle 6

2.1 Example of a binary decision tree 13
2.2 Comprehensible interpretable decision set 13
2.3 Feature importance of a linear model 15
2.4 Model explanation problem 17
2.5 Outcome explanation problem 17
2.6 LIME explains local behavior with feature importance 19
2.7 Toy example to present intuition for LIME 20
2.8 Genetic neighborhood data generation of LORE vs. uniformly random data generation 22

3.1 Difference recognition illustration of two datasets for black box A and B, used as a template for our running example 26
3.2 Another difference recognition illustration of two datasets for black box A and B 27
3.3 Difference classification problem 28
3.4 Process and involved components for training the diff-classifier of DiRo2C 29
3.5 Running example of a synthetic classification dataset for black box A and a manipulated dataset for black box B 30
3.6 Trained black box classifier A and B with the corresponding decision boundaries based on the running example dataset 31
3.7 Decision tree of trained binary black box classifier A and manipulated binary black box classifier B 32
3.8 Generated global synthetic datasets for training a binary and a multiclass diff-classifier 33
3.9 Global real data for training a binary and a multiclass diff-classifier . . . 33
3.10 Selected local real data for training a binary and a multiclass diff-classifier 35

4.1 Examples of a crossover and a mutation operation 43
4.2 Generated neighborhood datasets with the initial located instance to be recognized (Seed 1) 45

4.3 Generated neighborhood datasets with the instance to be recognized in the right bottom corner (Seed 2) 46
4.4 Generated neighborhood dataset with the instance to be recognized in the center (Seed 3) 47
4.5 Data density of the different local and global data approaches (Seed 1) . . 48
4.6 Data density of the local genetic neighborhood data approach examples . 49

5.1 Decision tree of the diff-classifier for analyzing the weak spot of the genetic neighborhood approach for DiRo2C 65

6.1 Running example: Datasets generated by the genetic neighborhood approaches with the initial located to be recognized instance (Seed 1) 71
6.2 Data density of both local genetic neighborhood data approaches (Seed 1) 72
6.3 Running example: Datasets generated by the genetic neighborhood approaches with the instance to be recognized in the right bottom corner (Seed 2) . . 74
6.4 Data density of both local genetic neighborhood data approaches (Seed 2) 75
6.5 Running example: Datasets generated by the genetic neighborhood approaches with the instance to be recognized in the center (Seed 3) 76
6.6 Data density of both local genetic neighborhood data approaches (Seed 3) 77
6.7 Example of a random Gaussian quantiles classification dataset for black box A and a manipulated dataset for black box B 78
6.8 Trained SVM black box classifier A and B based on the Gaussian quantiles dataset with the corresponding decision boundaries 79
6.9 Gaussian example: Datasets generated by the genetic neighborhood approaches with the instance to be recognized in the center (Seed 1) 80
6.10 Data density of both local genetic neighborhood data approaches (Seed 1) . 81
6.11 Gaussian example: Datasets generated by the genetic neighborhood approaches with the instance to be recognized in the corner (Seed 2) 82
6.12 Data density of both local genetic neighborhood data approaches (Seed 2) 83
6.13 Running example: Binary and multiclass global diff-dataset generated by using the modified genetic neighborhood approach and various instances to be recognized 84
6.14 Data density of the global modified genetic neighborhood dataset based on the running example 85

7.1 Running example of a synthetic classification dataset for black box A and a manipulated dataset for black box B 88
7.2 Trained black box classifier A and B with the corresponding decision boundaries based on the running example dataset 89
7.3 Decision tree of the trained binary black box classifier A and manipulated binary black box classifier B 90
7.4 Diff-classifiers trained by the global synthetic data approach based on the running example dataset 90

7.5 Decision trees of the diff-classifiers trained by the global synthetic data approach based on the running example dataset 91
7.6 Diff-classifiers trained by the global real data approach based on the running example dataset 91
7.7 Decision trees of the diff-classifiers trained by the global real data approach based on the running example dataset 92
7.8 Diff-classifiers trained by the local real data approach based on the running example dataset (Seed 1) 92
7.9 Decision trees of the diff-classifiers trained by the local real data approach based on the running example dataset (Seed 1) 93
7.10 Global diff-classifiers trained by the local real data approach based on the running example dataset and various seeds 93
7.11 Decision trees of the diff-classifiers trained by the global local real data approach based on the running example dataset 94
7.12 Diff-classifiers trained by the local genetic neighborhood data approach based on the running example dataset (Seed 1) 95
7.13 Decision trees of the diff-classifiers trained by the modified genetic neighborhood data approach based on the running example dataset (Seed 1) . . . 95
7.14 Global diff-classifiers trained by the concatenated local genetic neighborhood diff-datasets based on the running example dataset 96
7.15 Decision trees of the global diff-classifiers trained by the concatenated genetic neighborhood diff-datasets based on the running example dataset 96
7.16 Diff-classifiers trained by the local modified genetic neighborhood data approach based on the running example dataset (for Seed 1) 97
7.17 Decision trees of the diff-classifiers trained by the modified genetic neighborhood data approach based on the running example dataset (for Seed 1) 98
7.18 Global diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the running example dataset 99
7.19 Decision trees of the global diff-classifiers trained by the concatenated modified genetic neighborhood diff-datasets based on the running example dataset 100
7.20 Diagonal example of a synthetic classification dataset for black box B and a manipulated dataset for black box A 102
7.21 Trained black box classifier A and B with the corresponding decision boundaries based on the diagonal example dataset 103
7.22 Global diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the diagonal example dataset 105
7.23 Data density of the global modified genetic neighborhood dataset based on the diagonal example 105
7.24 Decision tree of the global binary diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the diagonal example 106

7.25 Decision tree of the global multiclass diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the diagonal example 109
7.26 Decision tree of black box A using the adult dataset for DMS1 111
7.27 Decision tree of black box B using the adult dataset for DMS1 112
7.28 Decision tree of the global binary diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the adult dataset for DMS1 113
7.29 Decision tree of the global multiclass diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the adult dataset for DMS1 114
7.30 Unlimited decision tree of black box A using the adult dataset for DMS1 115
7.31 Unlimited decision tree of black box B using the adult dataset for DMS1 . 116
7.32 Decision tree of the global binary diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the unlimited adult dataset for DMS1 117
7.33 Decision tree of black box B using the adult dataset for DMS2 118
7.34 Decision tree of the global binary diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the adult dataset for DMS2 118
7.35 Decision tree of the global multiclass diff-classifiers trained by the concatenated local modified genetic neighborhood diff-datasets based on the adult dataset for DMS2 119

List of Tables

5.1 Continous feature information of the adult dataset 54
5.2 Discrete feature information of the adult dataset 54
5.3 Continous feature information of the bank marketing dataset 55
5.4 Discrete feature information of the bank marketing dataset 55
5.5 Continous feature information of the credit approval dataset 56
5.6 Discrete feature information of the credit approval dataset 56
5.7 Diff-classifier performance results 64

6.1 Genetic neighborhood approaches: Comparison of the diff-classifier performance results 86

List of Algorithms

3.1 $BuildDiffDataset(b_A, b_B, X_{Diff}, Compare)$ 36

3.2 $CompareBinary(Y_A, Y_B)$ 36

3.3 $CompareMulticlass(Y_A, Y_B)$ 37

4.1 $LORE(x, b, N)$ 40

4.2 $GeneticNeigh(x, fitness, b, N, G, pc, pm)$ 41

4.3 $DiRo2C\_GN(x, b_A, b_B, N)$ 43

6.1 $DiRo2C\_MGN(x, b_A, b_B, N)$ 69

6.2 $ModifiedGeneticNeigh(x, modifiedFitness, b_A, b_B, N, G, pc, pm)$ . . . 69

Acronyms

ALE Accumulated Local Effects. 18

CART Classification And Regression Tree. 6, 27, 30, 56, 88, 104, 109, 122, 123

COMPAS Correctional Offender Management Profiling for Alternative Sanctions. 2

CSV Comma-Separated Values. 55

DiRo2C Difference Recognition of 2 Classifiers. 3, 5, 6, 8, 9, 11, 18, 23, 25–27, 33, 35, 36, 39, 42, 45–47, 57, 61, 71, 74, 76, 80, 82, 84, 101, 103, 120–124

GAMs Generalized Additive Models. 18

GDPR General Data Protection Regulation. 2

ICE Individual Conditional Expectation. 17

IME Interactions-based Method for Explanation. 20, 23

IS Information System. 5

LIME Local interpretable model-agnostic explanations. 18–20, 22, 23, 56, 57, 121, 123

LORE LOcal Rule-based Explanations. 3, 5, 6, 18, 20, 21, 23, 33, 39–42, 49, 52, 67–69, 88, 121–123

MES Model Explanation System. 20

ML Machine Learning. 1–3, 5, 12, 14–16, 18, 123, 124

NN Neural Network. 12, 14, 15

PALM Partition Aware Local Model. 18

PDP Partial Dependency Plot. 17, 18

Pearson CC Pearson Correlation Coefficient. 7, 61–64, 84, 86, 121

SHAP SHapley Additive exPlanations. 20

SVM Support Vector Machine. 12, 16, 77

XAI Explainable Artificial Intelligence. xi

Bibliography

[AZ19] Daniel W. Apley and Jingyu Zhu. Visualizing the effects of predictor variables in black box supervised learning models. *arXiv preprint arXiv:1612.08468v2*, 2019.

[Bal94] Shumeet Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical report, Carnegie Mellon University, Pittsburgh, PA, 06 1994.

[BFM00] T. Bäck, D.B. Fogel, and Z. Michalewicz. *Evolutionary Computation 1: Basic Algorithms and Operators*. Basic algorithms and operators. CRC press, 2000.

[BFOS84] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. The Wadsworth statistics/probability series. Wadsworth & Brooks/Cole Advanced Books & Software, Monterey, CA, 1984.

[CF16] Luis M. Candanedo and Véronique Feldheim. Accurate occupancy detection of an office room from light, temperature, humidity and CO 2 measurements using statistical learning models. *Energy and Buildings*, 112:28–39, 01 2016.

[CGH09] Anne Cleven, Philipp Gubler, and Kai M. Hüner. Design alternatives for the evaluation of design science research artifacts. In *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology - DESRIST '09*. ACM Press, 2009.

[CHL05] José Ramón Cano, Francisco Herrera, and Manuel Lozano. Stratification for scaling up evolutionary prototype selection. *Pattern Recognition Letters*, 26(7):953–963, 2005.

[CJ20] Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (MCC) over f1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21(1), 01 2020.

[CLG+15] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for HealthCare. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 08 2015.

[DB18] Jaka Demar and Zoran Bosni. Detecting concept drift in data streams using model explanation. *Expert Systems with Applications*, 92(C):546–559, 02 2018.

[DG17] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[DGH10] Joaquín Derrac, Salvador García, and Francisco Herrera. A survey on evolutionary instance selection and generation. *International Journal of Applied Metaheuristic Computing*, 1(1):60–92, 01 2010.

[DVK17] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608v2*, 2017.

[Esh91] Larry J. Eshelman. The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In GREGORY J.E. RAWLINS, editor, *Foundations of Genetic Algorithms*, volume 1 of *Foundations of Genetic Algorithms*, pages 265–283. Elsevier, 1991.

[FDRG+12] Félix-Antoine Fortin, François-Michel De Rainville, M.A. Gardner, Marc Parizeau, and Christian Gagné. Deap: Evolutionary algorithms made easy. *Journal of Machine Learning Research, Machine Learning Open Source Software*, 13:2171–2175, 07 2012.

[FGL12] Johannes Fürnkranz, Dragan Gamberger, and Nada Lavrač. *Foundations of Rule Learning*. Springer Berlin Heidelberg, 2012.

[Fre14] Alex A. Freitas. Comprehensible classification models. *ACM SIGKDD Explorations Newsletter*, 15(1):1–10, 03 2014.

[Fri01] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 10 2001.

[FW98] Eibe Frank and Ian H. Witten. Generating accurate rule sets without global optimization. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, page 144–151, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[FZL12] Yifan Fu, Xingquan Zhu, and Bin Li. A survey on instance selection for active learning. *Knowledge and Information Systems*, 35(2):249–283, 06 2012.

[GF17] Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine*, 38(3):50–57, 10 2017.

[GKBP15] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *Journal of Computational and Graphical Statistics*, 24(1):44–65, 01 2015.

[GMCR04] João Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. Learning with drift detection. In *Advances in Artificial Intelligence – SBIA 2004*, pages 286–295. Springer Berlin Heidelberg, 2004.

[GMR+18] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. Local rule-based explanations of black box decision systems. *arXiv preprint arXiv:1805.10820v1*, 2018.

[GMR+19] 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*, 51(5):1–42, 01 2019.

[Gol89] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., USA, 1st edition, 1989.

[Han06] David J. Hand. Classifier technology and the illusion of progress. *Statistical Science*, 21(1), 02 2006.

[Han12] David J. Hand. Assessing the performance of classification methods. *International Statistical Review*, 80(3):400–414, 08 2012.

[Has90] Trevor Hastie. *Generalized additive models*. Chapman and Hall, London New York, 1990.

[HDM+11] Johan Huysmans, Karel Dejaeger, Christophe Mues, Jan Vanthienen, and Bart Baesens. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decision Support Systems*, 51(1):141–154, 04 2011.

[Hev07] Alan R. Hevner. The three cycle view of design science research. *Scandinavian Journal of Information Systems*, 19(2):87, 2007.

[HM15] Mohammad Hossin and Sulaiman M.N. A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, 5(2):01–11, 03 2015.

[HMG+14] Stefan Haufe, Frank Meinecke, Kai Görgen, Sven Dähne, John-Dylan Haynes, Benjamin Blankertz, and Felix Bießmann. On the interpretation of weight vectors of linear models in multivariate neuroimaging. *NeuroImage*, 87:96–110, 02 2014.

[HMPR04] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *Management Information Systems Quarterly*, 28(1):75, 2004.

[Hol92] John H. Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. MIT Press, Cambridge, Mass, 1992.

[HPB+14] Andreas Henelius, Kai Puolamäki, Henrik Boström, Lars Asker, and Panagiotis Papapetrou. A peek into the black box: exploring classifiers by randomization. *Data Mining and Knowledge Discovery*, 28(5-6):1503–1529, jul 2014.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer New York, 2009.

[KCK20] Sourabh Katoch, Sumit Singh Chauhan, and Vijay Kumar. A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, 80(5):8091–8126, 10 2020.

[KHB+17] Joshua Kroll, Joanna Huey, Solon Barocas, Edward Felten, Joel Reidenberg, David Robinson, and Harlan Yu. Accountable algorithms. *University of Pennsylvania Law Review*, 165:633–705, 02 2017.

[Kin18] John Kingston. Artificial intelligence and legal liability. *arXiv preprint arXiv:1802.07782v1*, 2018.

[Koh95] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence (IJCAI)*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[KRS14] Been Kim, Cynthia Rudin, and Julie Shah. The bayesian case model: A generative approach for case-based reasoning and prototype classification. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 1952–1960, Cambridge, MA, USA, 2014. MIT Press.

[KW17] Sanjay Krishnan and Eugene Wu. PALM: Machine Learning Explanations For Iterative Debugging. In *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics*. ACM, 05 2017.

[LBL16] Himabindu Lakkaraju, Stephen Bach, and Jure Leskovec. Interpretable decision sets: A joint framework for description and prediction. In *KDD : proceedings. International Conference on Knowledge Discovery & Data Mining*, volume 2016, pages 1675–1684, 08 2016.

[LCG12] Yin Lou, Rich Caruana, and Johannes Gehrke. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '12*. ACM Press, 2012.

[Lip17] Zachary C. Lipton. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490v3*, 2017.

[LK20] Arnaud Van Looveren and Janis Klaise. Interpretable counterfactual explanations guided by prototypes. *arXiv preprint arXiv:1907.02584v2*, 2020.

[LL17] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017.

[LRMM15] Benjamin Letham, Cynthia Rudin, Tyler H. McCormick, and David Madigan. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 09 2015.

[LZL14] Ning Lu, Guangquan Zhang, and Jie Lu. Concept drift detection via competence models. *Artificial Intelligence*, 209:11–28, 04 2014.

[Mal19] Gianclaudio Malgieri. Automated decision-making in the EU member states: The right to explanation and other "suitable safeguards" in the national legislations. *Computer Law & Security Review*, 35(5):105327, 10 2019.

[MCR14] Sérgio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 06 2014.

[Mil19] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 02 2019.

[Mol20] Christoph Molnar. Interpretable machine learning, 2020. `https://christophm.github.io/interpretable-ml-book/`.

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA, 2013. Curran Associates Inc.

[MVED17] Dmitry M. Malioutov, Kush R. Varshney, Amin Emad, and Sanjeeb Dash. Learning interpretable classification rules with boolean compressed sensing. In *Studies in Big Data*, pages 95–121. Springer International Publishing, 2017.

[Pow08] David Powers. Evaluation: From precision, recall and f-factor to roc, informedness, markedness & correlation. *Machine Learning Technologies*, 2, 01 2008.

[Qui86] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[Qui87a] J.R. Quinlan. Generating production rules from decision trees. *Proceedings of the 10th International Joint Conference on Artifical Intelligence - Volume 1*, 1987.

[Qui87b] J.R. Quinlan. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27(3):221–234, 09 1987.

[Ras18] Sebastian Raschka. MLxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. *Journal of Open Source Software*, 3(24):638, 04 2018.

[Rij79] C. J. Rijsbergen. *Information retrieval*. Butterworths, London Boston, 1979.

[Riv87] Ronald L. Rivest. Learning decision lists. *Machine Learning*, 2(3):229–246, 11 1987.

[RSG16a] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. In *ICML Workshop on Human Interpretability in Machine Learning (WHI)*, 2016.

[RSG16b] Marco Túlio 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*, KDD '16, page 1135–1144, New York, NY, USA, 2016. Association for Computing Machinery.

[RSG18] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 04 2018.

[Sha53] Loyd S Shapley. "a value for n-person games". *Contributions to the Theory of Games 2.28*, pages 307–317, 1953.

[ŠK10] Erik Štrumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, 11:1–18, 2010.

[ŠKŠ09] Erik Štrumbelj, Igor Kononenko, and Marko Robnik Šikonja. Explaining instance classifications with interactions of subsets of feature values. *Data & Knowledge Engineering*, 68(10):886–904, 10 2009.

[SP18] Andrew Selbst and Julia Powles. "meaningful information" and the right to explanation. In *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 48–48, New York, NY, USA, 02 2018. PMLR.

[TSK06] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Pearson Education, 2006.

[Tsy04] Alexey Tsymbal. The problem of concept drift: definitions and related work. *Computer Science Department, Trinity College Dublin*, 106(2):58, 2004.

[Tur16] Ryan Turner. A model explanation system. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 09 2016.

[VPHB12] John Venable, Jan Pries-Heje, and Richard Baskerville. A comprehensive framework for evaluation in design science research. In *Lecture Notes in Computer Science*, volume 7286, pages 423–438. Springer Berlin Heidelberg, 05 2012.

[WFH11] Ian H. Witten, Eibe Frank, and Mark A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Elsevier, 2011.

[WHC+16] Geoffrey I. Webb, Roy Hyde, Hong Cao, Hai Long Nguyen, and Francois Petitjean. Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4):964–994, 04 2016.

[WMR18] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harvard Journal of Law & Technology*, 31:841–887, 04 2018.

[WO06] Shuning Wu and Sigurdur Olafsson. Optimal instance selection for improved decision tree induction. In *IIE Annual Conference. Proceedings*, page 1. Institute of Industrial and Systems Engineers (IISE), 2006.

[Won15] Tzu-Tsung Wong. Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation. *Pattern Recognition*, 48(9):2839–2846, 09 2015.

[WR15] Fulton Wang and Cynthia Rudin. Falling Rule Lists. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pages 1013–1022, San Diego, California, USA, 05 2015. PMLR.

[WY17] Tzu-Tsung Wong and Nai-Yu Yang. Dependency analysis of accuracy estimates in k-fold cross validation. *IEEE Transactions on Knowledge and Data Engineering*, 29(11):2417–2427, 11 2017.

[WY20] Tzu-Tsung Wong and Po-Yang Yeh. Reliable accuracy estimates from k-fold cross validation. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1586–1594, 08 2020.