UPPSALA
UNIVERSITET

UPTEC IT 23037
Examensarbete 30 hp
Oktober 2023

User Preference-Based Evaluation of Counterfactual Explanation Methods

Muhammad Zain Akram

Civilingenjörsprogammer i informationsteknologi

UPPSALA
UNIVERSITET

User Preference-Based Evaluation of Counterfactual Explanation Methods

Muhammad Zain Akram

Abstract

Explainable AI (XAI) has grown as an important field over the years. As more complicated AI systems are utilised in decision-making situations, the necessity for explanations for such systems is also increasing in order to ensure transparency and stakeholder trust. This study focuses on a specific type of explanation method, namely counterfactual explanations. Counterfactual explanations provide feedback that outlines what changes should be made to the input to reach a different outcome. This study expands on a previous dissertation in which a proof-of-concept tool was created for comparing several counterfactual explanation methods. This thesis investigates the properties of counterfactual explanation methods along with some appropriate metrics. The identified metrics are then used to evaluate and compare the desirable properties of the counterfactual approaches. The proof-of-concept tool is extended with a properties-metrics mapping module, and a user preference-based system is developed, allowing users to evaluate different counterfactual approaches depending on their preferences. This addition to the proof-of-concept tool is a critical step in providing field researchers with a standardised benchmarking tool.

Teknisk-naturvetenskapliga fakulteten

Uppsala universitet, Utgivningsort Uppsala

Handledare: Vandita Singh, Kristijonas Cyras Ämnesgranskare: Maria Andreina Francisco Rodriguez

Examinator: Lars-Åke Nordén

Sammanfattning

Förklarbar AI (XAI) har vuxit som ett viktigt område under åren. Eftersom mer komplicerade AI-system används i beslutsfattande situationer ökar också behovet av förklaringar till sådana system för att säkerställa transparens och intressenternas förtroende. Denna studie fokuserar på en specifik typ av XAI-metoder, nämligen kontrafaktiska förklaringsmetoder. Kontrafaktiska förklaringsmetoder ger feedback som beskriver vilka förändringar bör göras till inputen för att nå ett annat resultat. Denna studie utvidgar en tidigare avhandling där ett proof-of-concept-verktyg skapades för att jämföra flera kontrafaktiska förklaringsmetoder. Denna avhandlingsstudien undersöker de egenskaperna hos kontrafaktiska förklaringsmetoder tillsammans med några lämpliga mått. De identifierade måtten används sedan för att utvärdera och jämföra de önskvärda egenskaperna hos de kontrafaktiska metoder. Proof-of-concept-verktyget utökas med en egenskaps-metrics mappningsmodul, och ett användarpreferensbaserat system utvecklas, vilket gör det möjligt för användare att utvärdera olika kontrafaktiska metoder beroende på deras preferenser. Detta tillägg till proof-of-concept-verktyget är ett viktigt steg för att förse fältforskare med ett standardiserat benchmarkingverktyg.

Acknowledgements

I would like to thank Ericsson for providing me the opportunity to work on my master's thesis with them. I would also like to thank my Ericsson supervisors, Kristijonas Cyras and Vandita Singh, who believed in me and supported me during the thesis work. Their assistance with the thesis helped me overcome the difficulties.

In addition, I would like to thank my Uppsala University supervisor, Maria Andreina Francisco Rodriguez, for her guidance and insights during the thesis process.

Muhammad Zain Akram
Uppsala University, August 2023

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Objectives . . . 2
- 1.3 Limitations . . . 3
- 1.4 Disposition . . . 3

2 Literature Review **5**
- 2.1 Background . . . 5
  - 2.1.1 Black-box vs White-box Models . . . 5
  - 2.1.2 Explainable AI . . . 6
  - 2.1.3 Key Terminology in XAI . . . 6
  - 2.1.4 Classification of XAI Methods . . . 7
  - 2.1.5 Post-hoc Explanation Methods . . . 8
- 2.2 Counterfactual Explanations . . . 9
  - 2.2.1 Example Scenario . . . 9
  - 2.2.2 Counterfactual Explanation Methods & their Desired Properties 10
  - 2.2.3 Evaluation of Counterfactual Explanation Methods . . . 13
- 2.3 Summary . . . 13

3 Description of the Existing Proof of Concept **15**
- 3.1 The Dataset Stage . . . 15
- 3.2 The Machine Learning Model Stage . . . 15
- 3.3 The Counterfactual Explanation Methods Stage . . . 16
  - 3.3.1 Baseline Method . . . 16
  - 3.3.2 DiCE . . . 17
- 3.4 The Result stage . . . 17

4 Evaluation Metrics **19**
- 4.1 Distance metrics . . . 19
- 4.2 Other Metrics . . . 20
- 4.3 User Preferences . . . 22

5 Related Work **24**
- 5.1 CARLA: Counterfactual And Recourse Library . . . 24
  - 5.1.1 Documentation . . . 26
  - 5.1.2 Dependencies . . . 26
  - 5.1.3 Stability and Usability . . . 27
- 5.2 Tool Comparison and Integration Decision . . . 27

6 System Design & Implementation **28**
6.1 System Design 28
6.1.1 User Preference-Based System 29
6.1.2 Properties & Metrics Mapping 30
6.2 Implementation 30
6.2.1 Datasets 31
6.2.2 ML-Models and Counterfactual Explanation Methods 31
6.2.3 Metrics Functions 31
6.2.4 Weights Calculation 33

7 Results And Discussion **34**
7.1 Experiments 34
7.2 Discussion 41

8 Conclusions **43**
8.1 Challenges & Limitations 43
8.2 Future Work 44

1 Introduction

Over the last decade, technological advancements have resulted in enormous growth in the Artificial Intelligence (AI) sector, with AI now being used in various sectors of our society, from self-driving cars [36, 9] in the transportation industry to disease prediction [20] in the healthcare industry. The performance of AI systems has greatly improved as a result of these advancements, but the systems have also become more complex and difficult to interpret, and are now considered black boxes [16]. This has given rise to research into the trustworthiness of AI systems, with the goal of designing systems that are more transparent and human interpretable [40]. For simple tasks, this can be accomplished by employing simple models that are simple to interpret and produce satisfactory results. However, tasks requiring complex models will face the issue of explainability.

Model understanding is not required for all applications, such as recommendation systems or settings that do not require human intervention [9]. In certain circumstances, there may be minimal or no consequences for unacceptable outcomes [9]. It is also probable that the problem has been thoroughly researched, or the models used have been extensively validated in real-world applications. The emphasis is placed in higher stakes settings where the models' decisions will have an impact on people's lives, such as in medical applications [10]. Auxiliary criteria such as model fairness become more important in these high stakes use cases [9]. This is why explainable AI has grown in importance as a research field.

Explainable AI (XAI) is a field which focuses on explaining to the users why and how an AI system makes decisions [40]. The goal of XAI is to increase transparency, trust and accountability in AI systems by allowing humans to comprehend the reasoning behind the decisions made by AI models. There are numerous explanation methods available to users that provide merely the why, i.e. the reason for a certain outcome, however this is not always sufficient to comprehend how to alter the outcome [15]. Counterfactual explanation methods are a type of explanation method that provides users with feedback on how the outcome of an AI system might be modified to achieve a more desirable result [15].

1.1 Motivation

Given that XAI is being used to explain a variety of complicated models in high-stakes domains [26], it is vital to verify that the explanations given by these approaches are trustworthy. While the purpose of explanation techniques is to give insights into the decision-making process of complicated models, their inner workings may not always be entirely interpretable. This brings the issue of believing the explanations. Using

numerous methods to generate explanations for the same situation is one solution to this challenge. However, while AI practitioners can get a cohesive understanding of model behaviour if various approaches give consistent explanations, this is not always the case. There may be times when explanations given by different methods contradict one other [26]. As a result, there is a significant demand for a standardised benchmarking platform that assures approaches can be compared in a transparent and meaningful setting [42]. Researchers must be able to quickly compare their suggested techniques to the vast array of currently accessible methods, and ensure that they are utilising the correct method for the situation at hand.

1.2 Objectives

This thesis study builds on a previous dissertation [52] in which a proof of concept (POC) tool was designed to perform a specific task: comparison of counterfactual explanation methodologies.

The purpose of this thesis is first to identify what properties exist for counterfactuals. This requires studying literature to determine what properties may be associated with counterfactual explanation approaches and whether there are any desirable properties. The term "properties" refers to the particular characteristics that these counterfactual explanations should possess in order to be considered effective and informative for their intended purpose. The next step is to determine how these desirable properties might be evaluated. This involves reviewing literature to see what different measures are available for evaluating counterfactual explanation methods. Therefore, it is critical to understand how counterfactual methods differ in terms of how they are generated and what specific properties they take into account. Following the selection of desirable properties and metrics, the task at hand is to specify mapping rules, i.e. which property could be evaluated by which measure. This mapping can subsequently be used to expand the flexibility of the POC tool by introducing an automated module of property and metric mapping. Another goal is to transform the POC into a user preference-based system, which will result in an updated output format for the POC.

Another aspect of this thesis study is to explore similar tools to the POC and identify the benefits and drawbacks of such tools, as well as whether or not the studied tools could be merged with the existing POC.

1.3 Limitations

The project is limited by its focus on explanations for machine learning models while excluding machine reasoning. The goal of the POC tool is to be able to compare different explanation methods for the same model and present the result in a more human understandable manner, so either machine learning or machine reasoning models could have been used, but machine learning models were deemed comparatively easier to integrate into the system pipeline, being readily available as off-shelf libraries. The dataset used for this work is publicly available and is a tabular dataset. This is due to a variety of factors, including the dataset's simplicity and interpretability, which requires significantly less pre-processing than more complex datasets and is also easier to explain to non-experts. Another reason is computational efficiency, which means that running experiments and training models necessitates fewer computational resources. Along with publicly available datasets, the experimentation was carried out on the 5G-Slicing use case, and the general nature of the data was tabular, thus the emphasis was on datasets with similar modalities as the telecom use case.

For this work, metrics for different types of explanation methods are investigated, namely attributive explanation methods and contrastive explanation methods, with an emphasis on counterfactual explanation methods, which are a type of contrastive explanation method. This is because the existing implementation of POC tool is based on metrics and properties for counterfactual explanations. Furthermore, the evaluation of explanation methods is limited by a sole focus on statistical methods. Statistical methods provide means to objective analysis to an extent, without the need for extensive domain knowledge as may be the case with other techniques that could be based on building knowledge graphs, etc. Also, it is easier to interpret for non-expert users, as general form of representation is quantifiable and discrete in nature.

1.4 Disposition

The report is divided into 8 sections. The Introduction section offers an overview of the project and introduces the project objectives and limitations. Section 2 provides a detailed background information for Explainable AI and counterfactuals explanation methodologies, laying the groundwork for the subsequent sections. The following section exclusively contains material deemed relevant for this project that was discovered during the project's literature review phase.

The theoretical foundation for the project is detailed in the 3rd and the 4th section. This comprises a summary section for the current components as well as theory for the components that are implemented. The related works are presented in section 5 where a comparable tool is discussed in detail. The 6th section includes an overview of the system design as well as an implementation part that details thoroughly how the new

module was developed. This part expands on the theory offered in the 4th section by presenting the implementation components and methods. Section 7 presents the findings of the experimentation for the new module. It also explains why the various tests were conducted and what the results mean. Finally, section 8 provides concluding notes that summarise the important results. Additionally, future research proposals and prospective areas for development are suggested.

2 Literature Review

Explainable Artificial Intelligence (XAI) has become an increasingly important field of research as the need for transparent and trustworthy AI systems grows [4], with many approaches and methodologies being investigated to make AI more interpretable. Some researchers are interested in varied approaches concerning the models, whereas others are interested in various forms of explanations. This section will provide background information on XAI and its importance. The definitions and methods used in XAI research will be explored. Furthermore, this section will delve into the specific area of counterfactual explanations and their potential benefits and challenges. Overall, this literature review aims to provide a thorough overview of the topic of XAI, with a particular emphasis on counterfactual explanations and their desirable characteristics, that could be quantified.

2.1 Background

This section establishes the groundwork for an extensive understanding of XAI by providing the necessary material needed to comprehend this thesis work. The key terms and concepts of XAI are addressed briefly. The categorisation of XAI approaches found in literature is also presented.

2.1.1 Black-box vs White-box Models

The demands for transparency in AI systems are increasing from different stakeholders as more black-box machine learning models are being used to make predictions for critical situations [46]. Although what exactly do black-box models imply?

In engineering and computing, a **black-box** is a device or system that, given an input, produces some output without exposing any information about its underlying functions [24]. Black-box machine learning models are those that are complicated in nature and whose internal logic is difficult for humans to grasp [3]. By not having any insights on the internal workings, it creates uncertainty regarding the decisions made by such models. One such example of these black-box models is Deep Neural Networks (DNNs), as they are complex in nature with huge parametric space.

White-box models are the opposite of black box models, which are transparent in nature [30]. The domain experts can usually understand these models as they have knowledge of models internal working through some way, it can be gradients or weights. Models that are based on patterns, rules, and decision trees are labeled as white-box models [32].

2.1.2 Explainable AI

Different researchers have discussed explainable AI in different ways. It has been described as a branch of AI, as well as a property of AI systems. As stated in the preceding section, there is inconsistency in the definitions of the terms explainability and interpretability, where they are sometimes used interchangeably and sometimes defined separately. As a result, several definitions of explainable AI exist.

D. Gunning in [17] defines explainable AI as systems that have the ability to explain their logic to users, allowing the user to obtain a grasp of the system's behaviour and offers awareness of its strengths and weaknesses. The author further claims that XAI will create a set of machine learning techniques, that will allow the users to comprehend and trust the systems [17]. This includes any strategy that minimises the complexity of the model and simplifies its outputs [4].

The focus of such AI systems is the end users since they rely on the predictions provided by these AI systems and hence need to understand the results generated by these systems. For self-driving cars, for example, the operator has to comprehend the autonomous vehicle's decision-making system in order to effectively deploy for future systems [17]. Depending on the target audience the requirement from such explainable systems might differ, for instance in the definition presented by D. Gunning, brings concepts such as trust and understandability but that might not be the only purposes for such systems for an audience, there can be fairness or even confidence [4]. If the intended audience includes domain specialists, they must be able to trust the model and derive insights from it, whereas end users may simply need to know if the predicted outcome by the model is fair. Keeping the audience in focus, Arrieta et al. [4] define XAI as one that produces insights or reasoning to make its work more transparent or easier to understand for a specific audience.

2.1.3 Key Terminology in XAI

The words *explainability* and *interpretability* are frequently used interchangeably in the literature on XAI. When it comes to model **interpretability**, different authors have related the word with various aspects of the model. For example, interpretability has been promoted as a condition for trust. According to Ribeiro et al. [48], an interpretable model is one that provides a qualitative understanding of the input and result. Whereas Arrieta et al. [4] define interpretability as a passive feature of a model that relates to the degree to which models are intelligible to people. **Explainability** relates to providing an explanation, which in the literature is characterised as an interface between people and the model, and offers an accurate approximation of the decision-making model that is understandable to humans [16].

Other terms used in the literature in the XAI sector include understandability and transparency. **Understandability** refers to the property of a model that is humanly comprehensible without the need for any understanding or explanation of its internal workings [4]. The term **transparency** relates to understandability and requires a model that can be understood on its own [4]. A model can have varying degrees of transparency. It can be considered at the model's overall level, at the level of individual components, or at the algorithm level [30].

2.1.4 Classification of XAI Methods

Over the years, the researchers have investigated and created a diverse range of XAI methodologies for various objectives and target audiences. According to the literature [58], there are five major groups into which XAI techniques may be classified. These include stage, scope, problem types, input and output formats. Figure 1 depicts a hierarchical design that divides the XAI approaches into distinct groups. This overview is based on Vilone and Longo's review research [58].

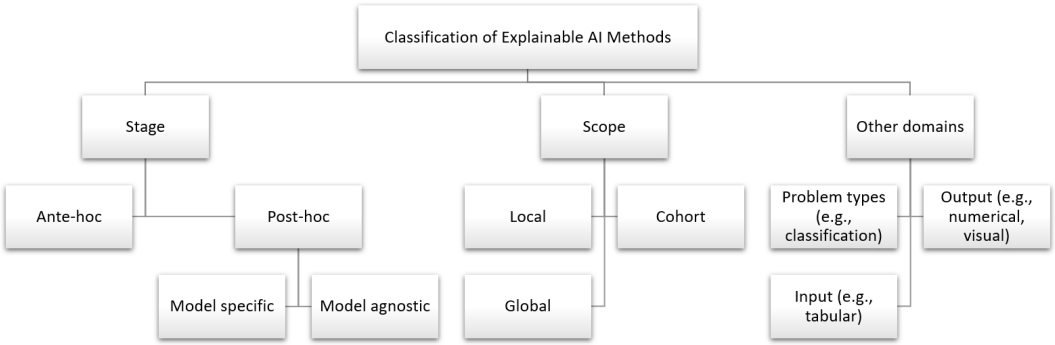

Figure 1 Classification of XAI methods based on [58].

Stage

The stage of explainability refers to the period in the process at which the model generates the explanation for the decision it provides [19]. The XAI methods considering the stage of explainability are divided into 2 categories; ante-hoc and post-hoc. Ante-hoc methods consider the explainability of the model during the design and development of the machine learning model itself [58]. These methods focus on creating inherently

interpretable models or designing features that facilitate explanation generation, even before the model is deployed. Post-hoc methods consist of an external model which is used to generate an explanation for the base model. The base models are often black-boxes [52], where explanations are generated by post-hoc methods once the predictions are made [19]. Furthermore, post-hoc approaches are classified as model-agnostic or model-specific. Model-specific methods can only be used for certain ML models, whereas model-agnostic methods can be applied to any sort of model.

Scope
The scope of explainability specifies the extent of an explanation that is produced by some XAI method [19]. In terms of explainability scope, the methods are divided into three categories, namely local, cohort, and global. The XAI methods that explain a single prediction instance of a model are referred to as local explanation methods. The methods that consider a sub-group of predictions to explain are cohort methods. The goal of global explanation methods is to make a model's complete inferential process transparent [58], allowing for interpretation of the full set of predictions.

Problem Types, Input and Output Formats
XAI methods might differ depending on the underlying problem, such as classification or regression. In terms of input data, the explainable models differ as well. Because the technique used by a model to categorise images might differ significantly from textual data, this can play a key role in the development of explainable methods algorithms. Similarly, various target audiences and situations may need distinct output formats from these methodologies. The most common forms of output for explanation methods recognised by Vilone and Longo [58] are numeric, rules, textual, visual and mixed.

2.1.5 Post-hoc Explanation Methods

As previously stated, post-hoc techniques can be either model-specific or model-agnostic. Post-hoc model-agnostic methods refers to explanation methods that can be applied to any type of machine learning model and provide explanations after the model has been trained. Furthermore, model-agnostic techniques can be divided into three categories: model simplification, feature importance, and local explanations [4, 57]. Feature importance methods identify the most important features that influence a particular decision. Shapley Values (SHAP) [34] is one example of such a methods. Local explanations methods only explain a single prediction of a model. Local explanation methods can further be divided into approximation and example-based methods. Lets take x as an example of prediction from the model that needs to be explained. Approximation-based methods sample new datapoints in the proximity of x and then can either fit a linear model or derive a rule set from them [57]. Local Interpretable Model Agnostic Ex-

plainer (LIME) [48] is an example of method that fits linear model and Anchors [49] are example of rule-based methods. Counterfactual explanation is an example-based technique that looks for data points around x that have a different prediction than x. The subsequent section presents counterfactual explanations in-depth, delving into specific details, methodologies and findings related to it.

2.2 Counterfactual Explanations

Counterfactual explanations have arisen as a significant approach in XAI because they provide a unique way of presenting explanations. These methods present alternative scenarios that result in a different outcome, giving information on the factors that influence the model's decision-making process. Counterfactuals have been in psychology for a long time. Lewis proposed a theoretical definition in 1973 [57, 28]. Wachter et al. [60] were among the first to propose the use of counterfactuals in machine learning.

A counterfactual is the smallest change in the input features that changes the prediction to another outcome. There are other terms used interchangeably for counterfactuals in the literature [57], such as contrastive explanation [7] and algorithmic recourse [23]. Other explanation methods, such as SHAP [34], provide feature importance, which states which features contribute the most to a specific outcome, whereas counterfactual explanations provide actionable feedback [57] that when applied to the features, can result in desired outcome.

2.2.1 Example Scenario

Let's consider an example case to better understand the counterfactual explanations. Consider a hospital that uses a machine learning model to predict the possibility of post-operative problems. This model is designed to predict if a certain patient will have an elevated risk of complications if they undergo a given procedure. The model takes as input certain patient information, in this example $\{$age, gender, BMI, blood glucose levels, and blood pressure$\}$, and outputs a binary classification,$\{$high risk, low risk$\}$, indicating whether the patient is at high or low risk of post-op problems. One of the hospital's doctors, John, applies this model to one of his patients and receives high risk as the model's output. Now John is wondering, 1) What features contributed to this particular outcome? 2) What should he advise his patient to do in order to get the desired result? In this scenario, the intended goal is that his patient has a low risk of problems following surgery. The first question can be addressed using any explanation technique that outputs the feature importance, such as "Blood sugar levels are too high," however this does not solve the issue at hand. Counterfactual explanations can be used to acquire a better understanding of the model's prediction. The aim is to find actionable insights and strategies that can help reduce John's patient's anticipated risk of problems. Multi-

ple counterfactuals are possible [60] as various factors may impact the outcome, leading to multiple ways to obtain the desired outcome. A few examples of hypothetical counterfactuals created by counterfactual explanation methods may be as follows: ”Reduce blood sugar levels by 5%” or ”Reduce BMI by 10%.” This is illustrated by Figure 2. This form of explanation not only offers a notion of which variables are contributing to the projected outcome, but it also provides actionable feedback that John can use to recommend to his patient so that the probability of difficulties post-op is minimal. The counterfactual explanation methods guarantee that if the recommended changes are implemented, the intended outcome will be obtained, provided that the model used for prediction has not changed [57].

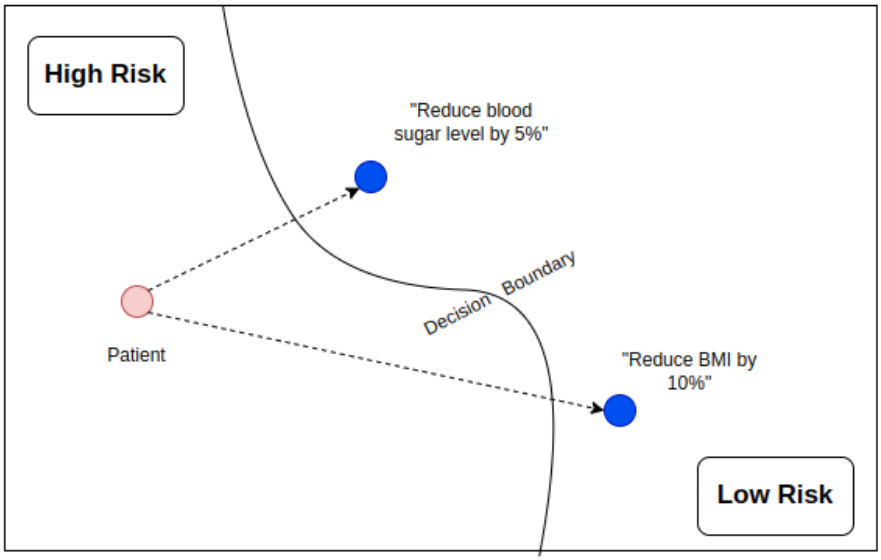

Figure 2 Counterfactual explanation examples for the patient with high-risk scenario.

2.2.2 Counterfactual Explanation Methods & their Desired Properties

The literature specifies certain desirable properties that a counterfactual should have in order to achieve its purpose [57, 60, 39] . Research in the development of counterfactual explanation methods has focused on generating counterfactuals that ensure some of these properties [15]. These methods consider the different needs and requirements of various applications. Molnar [38] refers to such properties as “requirements” that a counterfactual should fulfill.

As previously stated, Wachter et al. [60] proposed counterfactual explanations as an optimisation problem. There the objective is to minimise the distance between the counterfactual and the original data point given the counterfactual belongs to the desired

label class. Verma et al. [57] states that a counterfactual classified in the desired class is a valid counterfactual, resulting in the property *validity* for counterfactuals. Higher validity value is preferred [57]. Another property induced from the Wachter et al. [60] is of *proximity*. This property implies that the counterfactual should suggest a relatively small change to input features, that results in the desired outcome [57]. Referring back to the example shown in Figure 2, the two generated counterfactuals are on the correct side of the decision boundary, since the desired label class is $\{$low risk$\}$. This meets the validity requirement for both counterfactuals. For proximity, for example, the suggested change is to reduce sugar levels by 5%. However, reducing it by 10% may also work, but it is practical to make the smallest possible change. Lower proximity values are preferred [57]. Proximity is often referred to as similarity in literature.

Another desirable quality for counterfactuals is *sparsity* also referred to as *minimality* [15]. According to research, people prefer simple explanations over complicated ones [37]. Sparsity captures this and implies that counterfactuals should ideally suggest a change in only a few features. For example, a counterfactual can prescribe making small alterations to three feature inputs to reach the desired outcome, but this is not optimal when compared to a counterfactual that only suggests one small change.

When one feature in the dataset has a direct impact on another, then a causal relationship exists among those feature. Hence, one event causes another to occur [51]. A counterfactual should adhere to any established causal relations between input features in order to be practical [57]. This results in the property called *causality* for counterfactuals. Mahajan et al. [35] presented a method to generate counterfactuals using causal models which took into consideration the causal relationships among input features.

In their publication, Poyiadzi et al. [45] propose FACE: Feasible and Actionable Counterfactual Explanations, a counterfactual generating method with the aim to provide feasible and actionable counterfactuals. This method considers the underlying data distribution and use high-density pathways specified by density-weighted metrics. The paper [45] discusses state-of-the-art counterfactuals generation methods and presents the shortcomings of these methods. It also specifies two more counterfactual properties: *feasibility* and *actionability*. According to Verma et al. [57], input features can be mutable or immutable. The term actionability refers to the fact that the counterfactual suggests changes that are achievable, i.e. the immutable features are not modified. Consider the example presented in section 2.2.1, the immutable attributes are age and gender, implying that changes to these features are not actionable. The authors of FACE [45] refer to the property feasibility as to revealing feasible paths based on the shortest path between the counterfactual and the original instance. Mahajan et al. [35] introduce feasibility as a causal term and argue that it cannot be handled just through statistical restrictions. The authors consider a counterfactual feasible if the changes fulfill the causal model's constraints. Another criterion of counterfactual is that it should

be close to the training data and conform to observable correlations between the features [57]. This is known as *data manifold closeness*. A counterfactual should not be unreasonable, which means it should not be an outlier.

There are several methods that can generate multiple counterfactuals for a single instance. One such example of method is Diverse Counterfactual Explanations (DiCE) [39]. Mothilal et al. [39] emphasise *diversity* for counterfactuals as a crucial component for actionable counterfactuals. When presented with a wide range of alternatives, it becomes easier for the user to pick the action [57]. Consider the example scenario from section 2.2.1 once more. If the proposed counterfactuals were both similar, such as "decrease blood sugar levels by i) 5% ii) 10%," it would be less helpful than presenting "decrease blood sugar levels by 5%" and "decrease BMI by 10%." Because the second group is more diversified, it provides more options for changing the outcome.

The Figure 3 illustrates an overview of the desirable properties for counterfactuals discussed in this section.

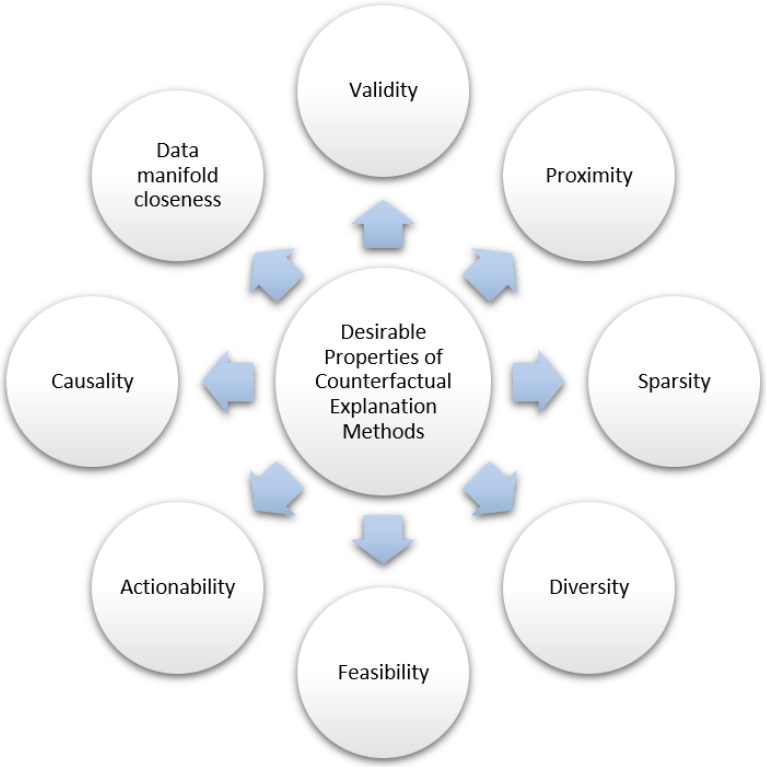

Figure 3 Desirable Properties of Counterfactual Explanation Methods.

2.2.3 Evaluation of Counterfactual Explanation Methods

The majority of counterfactual generating methods are evaluated based on the desirable properties of counterfactuals [57, 15]. To be able to compare different approaches and measure the quality of the explanations, counterfactual explanations must be evaluated. There are numerous metrics used in the literature to evaluate counterfactual explanations with respect to the desirable properties. Distance metrics like L1 norm, L2 norm, Mahalanobis distance, and so on are often used for evaluating properties that involve calculating the distance between counterfactuals and the original instance [57, 15]. Different papers, for example, use various forms of these distance metrics for proximity [57], such as computing the average weighted sum or dividing the distance by the median absolute deviation [57, 39, 60]. This is done for a variety of reasons, one of which is to deal with the diversity in range among different features. These distance measures are also used to assess sparsity [57, 15], where they may be utilised to quantify the magnitude with which features change. Metrics distance may also be used to assess the diversity of counterfactuals [57, 39]. There are also several scoring metrics developed to analyse counterfactuals, with Mahanjan et al. [35] proposing one to assess feasibility. This work [35] also discusses a metric for causality that measures whether or not changes in the counterfactual meet the causal relationship between features. Actionability and closeness to data can be captured for instance by measuring the average distance to the k-nearest data points [57] or measuring local outlier factor [22]. The number of counterfactuals in the desired class is frequently used to calculate validity [57, 15].

2.3 Summary

The literature review section provides a comprehensive overview of the relevant research papers and literature, establishing the basis for understanding the theoretical components and objectives of the project. The section begins by explaining various terms that are often utilised in the context of XAI. Terms like black-box and white-box models are defined, as are other terms like understandability, transparency, and interpretability, which are frequently cited in XAI literature. Different definitions of explainable AI found in the literature are described, with Arrieta et al. [4] presenting the concept of XAI with an emphasis on the target audience.

In addition, the section presents the categorisation of XAI approaches. The literature has a wide range of XAI approaches and methodologies, which has led to the categorisation of XAI methods into five broad groups [58]. These include stage, scope, problem types, input and output formats, and are discussed in further detail. Among them, the focus of this work is on counterfactuals explanation techniques, which fall under the wider category of post-hoc explanation methods.

Counterfactual explanation approaches provide users with actionable feedback [39] that,

when implemented, produces the desired result. An example scenario is presented in Section 2.2.1 to demonstrate the application and significance of counterfactual explanation approaches. The Sections 2.2.2 and 2.2.3 then go on to discuss the desired properties of counterfactuals as well as how to assess them. According to the literature [57, 60], counterfactuals must have particular properties in order to serve their purpose. Figure 3 shows the properties that are described in detail. To assess the quality of explanations and the effectiveness of such methods, desired properties of counterfactual explanation methods must be evaluated. The evaluation techniques can then be used to compare different explanation methods. The different evaluation metrics used in the literature to evaluate counterfactual explanation methods are also mentioned.

3 Description of the Existing Proof of Concept

The purpose of this section is to summarise the components of the existing POC and discuss the theoretical notions necessary to comprehend it. There are some differences between the pipeline implemented in the previous dissertation (version 1) [52] and the pipeline used as a starting point (version 2) for this study. This section will discuss the key components of both versions that are identical. The version 2 mentioned in this part does not include the new module added for this study, which is described in further detail in Section 6.

The existing POC was built as a pipeline comprised of four major building stages. These components include selecting a dataset, training a machine learning model, deploying counterfactual explanation methods, and analysing the results against the metrics of choice.

3.1 The Dataset Stage

In practice, any classification dataset may be used. For that the dataset must be pre-processed, which commonly includes data normalisation, data cleaning, feature selection, and so on. The datasets used for both version of the pipeline consisted of categorical and numerical tabular data. The dataset is split into training and testing datasets in an 80:20 ratio. After that, the dataset is used to train a classifier model.

3.2 The Machine Learning Model Stage

The machine learning (ML) model block in pipeline version 2 includes four classifiers: 1) Logistic Regression, 2) Random Forest Classifier,3) Decision Tree Classifier, and 4) XGBoost. Convolution Neural Networks (CNN) were also used in the pipeline version 1. For version 2 however, the CNN model was deemed unnecessary for two reasons: i) no image dataset was utilised for the experiments, and ii) the focus was not on comparing different classifier performances. All three machine learning models are supervised learning methods [29]. Logistic regression is used for binary or multi-class classification tasks. It fits a logistic function to input features and predicts using the calculated coefficients [29]. Decision tree is a rule-based method that uses a series of if-else conditions [29] depending on the feature values and makes predictions. It creates a tree-like model. Random forest is an ensemble method [29] which can be used for both classification and regression. This model combines output of multiple decision trees to reach a single result. XGBoost [6] is a gradient boosting algorithm that predicts using an ensemble of decision trees by minimising the objective function using boosting and regularisation strategies.

3.3 The Counterfactual Explanation Methods Stage

The pipeline version 1 employed four counterfactual explanation methods, which are as follows:

i) Counterfactual generation method proposed by Wachter et al. [60]. This is referred to as the baseline method in the dissertation.

ii) Diverse Counterfactual Explanations (DiCE) by Mothilal et al. [39].

iii) Contrastive Explanation Method (CEM) proposed by Dhurandhar et al [8].

iv) Counterfactual Explanations Guided by Prototypes proposed by Looveren [31].

The current explanation methods provide explanation for tree-based models, whereas the other two methods explain models that are primarily differential. This situation would result in an unfair comparison of all four methods. As a result, only i) and ii) are employed in pipeline version 2, since they were deemed sufficient for experimentation and testing the extended module.

3.3.1 Baseline Method

The method proposed by Wachter et al. [60] has laid the foundation for several additional counterfactual generating methods. The authors present this as an optimisation problem shown in Equation 1. The objective is to minimise the distance between the counterfactual represented as x' and the original instance x_i to the point where $f_\omega(x')$ equals the new target y'. The first term drives the classifiers output to be near the target class , while the second term pushes the counterfactual to be near the original instance. The $d(,)$ denotes a distance function, and the proposed function is the Manhattan distance weighted by the inverse Median Absolute Deviation (MAD). To find a close solution, x' is solved iteratively by maximising the parameter λ.

$$\arg\min_{x'} \max_{\lambda} \lambda(f_\omega(x') - y')^2 + d(x_i, x') \tag{1}$$

3.3.2 DiCE

DiCE is a counterfactual explanation method proposed by Mothilal et al. [39]. The suggested method aims to generate counterfactuals that are both feasible and diversified. This approach also introduces an optimisation problem with various constraints. Equation 2 presents the objective function described by Mothilal et al. [39]. Where $f(.)$ is the ML model, k denotes the total number of counterfactuals to be created, c_i is the counterfactual example and x denotes the original instance. The $yloss$ is a modified loss term from the Wachter method [60] that encourges $f(.)$ output to be as near to the desired class y as possible. $dist$ is a distance function that is similar to the Wachter method. It is defined as the mean of feature-wise L1 distances between x and c_i divided by MAD.The $dpp\_diversity$ is the diversity metric. Furthermore, the loss function is balanced using the hyper-parameters λ_1 and λ_2

$$\arg\min_{c_1,...c_k} \frac{1}{k}\sum_{i=1}^{k} yloss(f(c_i), y) + \frac{\lambda_1}{k}\sum_{i=1}^{k} dist(c_i, x) - \lambda_2\, dpp\_diversity(c_1, ...c_k) \quad (2)$$

3.4 The Result stage

A wide range of metrics are implemented in the pipeline's version 1 to evaluate the counterfactuals. Counterfactuals are evaluated based on four separate characteristics: whether they belong to the intended class, prediction probability, proximity, and direct and indirect path assessments. These are evaluated in relation to the counterfactual and the original instance. Different distance functions, such as L1 norm, L2 norm, Mahalanobis distance, and so on, are employed to measure proximity. Different scoring metrics, such as feature value score, change count score, and neighbourhood score, are also implemented. Some of the above mentioned metrics are also employed in pipeline version 2. Section 4 presents all of the metrics implemented in pipeline version 2 as well as the theory to understand them.

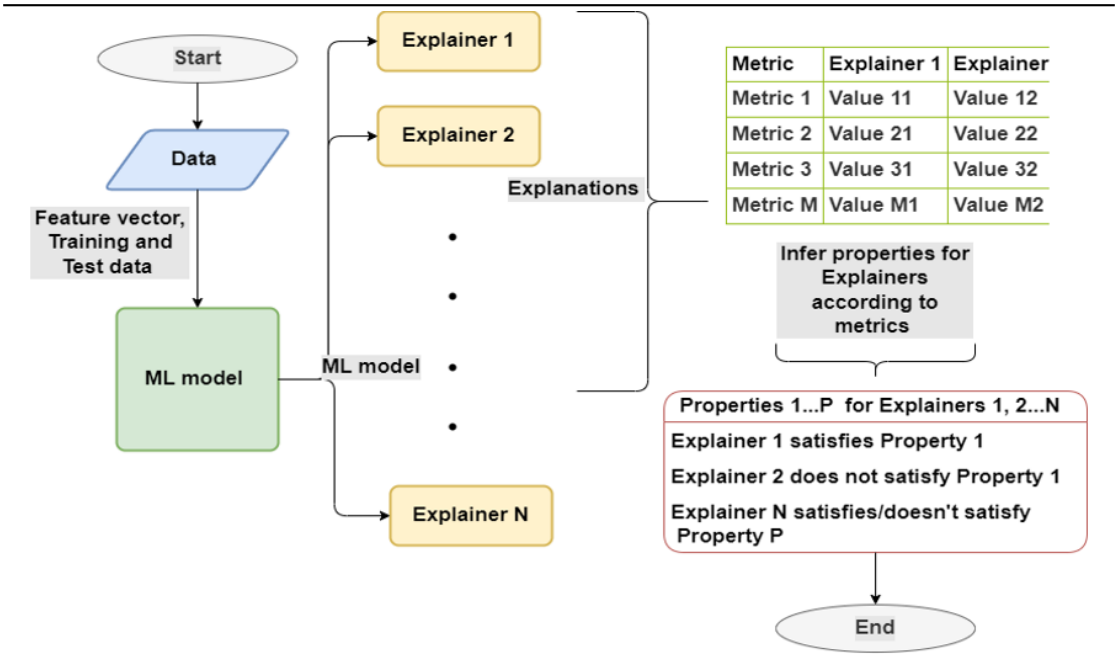

Figure 4 System Design pipeline based on [52].

Once the required metrics are chosen, the outcome is displayed to the user in a tabular style, with each metric used to assess each counterfactual created by different methods. Furthermore, the properties for the various approaches are inferred manually based on the metrics. Figure 4 shows the high-level system design for the pipeline. The main stages are shown and their details are summarised.

4 Evaluation Metrics

There are many metrics for evaluating counterfactual explanations. Section 2.2.3 discusses these. All of these metrics are used to evaluate some desirable property of counterfactual explanations. Different authors have employed different metrics to measure the same property, for example, different distance metrics for evaluating proximity [57], and they also used the same metric to measure different properties, for example, Mothilal et al [39] used feature wise distances using L1-norm for both proximity and diversity with some other constraints.

This section will focus on the metrics utilised in this work to assess counterfactual explanations for some desirable features. Section 6.1.2 presents the mapping of properties and measures. The purpose is to introduce the metrics so that the mapping of properties with metrics as well as the application of the mapping as a new module in the pipeline becomes easier to understand.

4.1 Distance metrics

Distance metrics are applied in all aspects of machine learning, whether it be computer vision and clustering tasks. In both supervised and unsupervised learning, distance measures are used to quantify similarity or assess proximity between datapoints. According to research [62, 63], an appropriate distance metric can help improve the trained model's performance. The following distance measurements were used in this study: L1-norm, L2-norm, Hamming distance and Cosine similarity.

Consider two vectors $x = (x_1, ..., x_n)$ and $y = (y_1, ..., y_n)$, then the mentioned metrics are calculated as following:

Manhattan Distance/ L1-norm

$$D(x, y) = \sum_{i=1}^{n} |x_i - y_i| \tag{3}$$

Euclidean Distance/ L2-norm

$$D(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{4}$$

Hamming Distance

$$D(x,y) = \sum_{i=1}^{n} \delta(x_i, y_i) \qquad where \quad \delta(x_i, y_i) = \begin{cases} 0, & \text{if } x_i = y_i \\ 1, & \text{if } x_i \neq y_i \end{cases} \tag{5}$$

Cosine Similarity

Cosine distance is represented as D_{cos} and Cosine similarity as D_{sim}.

$$D_{cos}(x,y) = 1 - D_{sim}(x,y) \tag{6}$$

$$D_{sim}(x,y) = \frac{\sum_{i=1}^{n} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}} \tag{7}$$

4.2 Other Metrics

This section presents metrics that are not distance metrics and are used for evaluating counterfactuals for some desirable properties. These include the Local Outlier Factor (LOF), the Validity Score, and the Median Absolute Deviation (MAD). It is important to point out that MAD is not employed as an evaluating measure on its own, but rather in conjunction with L1-norm for this work.

Local Outlier Factor (LOF)

Outliers are data objects that stand out from the rest of the dataset and do not follow the typical behaviour of the dataset. The process of identifying outliers in a dataset is known as anomaly detection [25]. The local outlier factor is a density-based anomaly detection technique. Breunig et al [5]. proposed LOF, which overcomes a major difficulty with other density-based outlier detection methods: finding outliers in variable density [25]. To thoroughly understand how the Local outlier factor is calculated for an object or a point, a few things must be addressed. The detailed explanation can be obtained by referring to the following resource [5]. These are as follows:

K-Distance and K-Neighbours:

Consider a point x; the k-distance is the distance between the point and its k^{th} nearest neighbour. Either Euclidean or Manhattan distance functions can be used. Every point whose distance from x is less than k-distance is considered a K-neighbour. The set of such points is denoted as $N_k(x)$

Reachability distance:

The reachability distance of the point x with respect to point o is defined as:

$$RD_k(x, o) = \max\{\text{k-distance}(o), d(x, o)\}$$

The distance function is denoted as $d(,)$.

Local Reachability Density:

The local reachability density is the inverse of the average reachability distance of x from its neighbours.

$$LRD_k(x) = 1/\left[\frac{\sum_{o \in N_k(x)} RD_k(x, o)}{|N_k(x)|}\right]$$

Local Outlier Factor:

The density of x can be determined by comparing it to the density of all datapoints in the neighbourhood.

$$LOF_k(x) = \frac{\sum_{o \in N_k(x)} \frac{LRD_k(o)}{LRD_k(x)}}{|N_k(x)|} \tag{8}$$

Assuming the point x is an outlier, then its LRD will be less than its neighbour's average LRD, resulting in a high LOF value. In contrast if x is not an outlier the LOF will be close to 1 since the average of LRD of neighbours will roughly equal to LRD of x.

Validity Score

A counterfactual is said to be valid if it changes its classification from the original class to the desired one, provided that the desired class is not the same as the original class. Validity score determines the fraction of counterfactuals that are classified in the desired class. Suppose K is the total number of counterfactuals generated by a method then the validity score is:

$$\text{Validity-Score} = \frac{\text{\# of CFs in desired class}}{K} \tag{9}$$

The number of counterfactuals in desired class can be determined by using the model and predicting the class probabilities for each counterfactual.

Median Absolute Deviation (MAD)

The Median Absolute Deviation (MAD) is a scale metric that can be used to summarise data variability. It is a robust measure of variability since it uses the median as an

estimate of the distribution's centre and the absolute difference rather than the squared difference [50]. It is calculated by taking the median of all absolute distances from the sample median.

Consider a dataset $X = \{x_1, x_2, ..., x_n\}$ with sample median as $\tilde{X} = \text{median}(X)$, then MAD can be calculated as follows:

$$MAD = \text{median}(|x_i - \tilde{X}|) \tag{10}$$

4.3 User Preferences

In utility theory, assumptions regarding a user's preferences over specific choice possibilities are established, which results in numerical values expressing the subjective preference ordering [11]. Utility functions can be used to mathematically model preferences and capture the user's preference among many options [59]. Utility theory has established a framework for modeling user preferences by assigning distinct choices a numerical value [12] termed utility. Preferences are divided into two types: *ordinal* preferences and *cardinal preferences*.

Consider an end user to be a decision-maker (DM) who states preferences among three choices, $\{$A,B,C$\}$. Preference information can be expressed in both cardinal and ordinal forms.

Ordinal Preferences

Ordinal preferences use a linear order to rank options in order of preference from best to worst, without any further information [14], such as assigning utility to distinct choices. If the DM picks A over C and C over B, then a utility function u can be represented as follows:

$$u(A) = 5, u(C) = 3, u(B) = 1$$

However, the above DM preferences can be represented equally effectively with another function v as long as the order of the preferences is the same; the numbers assigned are meaningless. The functions u and v are ordinally equivalent. As for ordinal preferences, only the order $u(A) > u(C) > u(B)$ is important.

In applied literature, ordinal information on preferences is derived from one of two sources: directly from DM or indirectly from cardinal information [14].

Cardinal Preferences

Cardinal preferences assign a numerical value to the options, allowing for quantitative evaluation of the preferences [14]. In contrast to ordinal, the numerical value assigned to preferences represents strength, therefore considering u and v as utility functions, if

u is defined as prior and v is defined as follows:

$$v(A) = 8, u(C) = 7, u(B) = 1$$

The difference between the assigned numbers is important as the difference between C and B for both functions varies by a lot, the functions are not cardinally equivalent.

5 Related Work

This section introduces tools for benchmarking evaluation methods that are comparable to the existing POC. OpenXAI [1] is an open-source framework for evaluating and benchmarking post hoc explanation methods. The OpenXAI tool aims to assess explanation methods based on properties: faithfulness, stability, and fairness. The tool includes both a synthetic dataset generator and a collection of real-world datasets. It also includes pre-trained models and twenty-two quantitative metrics for evaluating the properties listed above.

CARLA (Counterfactual And Recourse Library) [42] is a Python library that was created to provide baselines for comparing counterfactual explanation methods. The aim was to create a transparent and coherent framework for researchers to compare counterfactual explanation approaches on various datasets.

The library CARLA is investigated in more detail since the current POC also focuses on counterfactual explanation methods. The goal was to acquire insights from CARLA, identify its advantages and disadvantages, and decide whether any components can be integrated with the existing POC.

5.1 CARLA: Counterfactual And Recourse Library

As previously mentioned, CARLA is an open-source tool for benchmarking counterfactual explanation methods. It allows for a transparent and extensive comparison of 13 counterfactual methods with built-in machine learning models, and a standardised set of integrated evaluation measures and datasets. Furthermore, CARLA enables users to integrate new methods as well as plug in their custom black-box models into the tool.

Pawelczyk et al. [42] distinguish between independence,dependence and causal-based counterfactual explanation methods that are implementing in CARLA. The input features of the prediction model are assumed to be independent in independence-based methods and dependent in dependent methods. In causal-based approaches, the causal relationship between input features is considered, and such methods include causal modelling using structural equations or graphs [42].

Table 1 shows the methods implemented in CARLA that can be evaluated and used for benchmarking.

Independent	Dependent	Causal
AR (Actionable Recourse) [56]	C-CHVAE [43]	CR (Casual Recourse) [23]
CEM [7]	CLUE [2]	ROAR [55]
DiCE [39]	FACE [45]	
Growing Spheres [27]	REVISE [21]	
Wachter [60]		
FeatureTweak [54]		
FOCUS [33]		

Table 1: Counterfactual explanation methods implemented in CARLA [42].

Since the provided methods vary in their characteristics and focus on distinct qualities, a diverse set of assessment metrics is required to analyse the various approaches. CARLA offers six baseline assessment metrics. Table 2 briefly outlines the various measures.

Metrics	Description
Costs	The distance of the original instance to the counterfactual. L0, L1, L2, L-infinity distance measures.
Constraint Violation	Counts the number of times the counterfactual explanation method violates user-defined constraints.
yNN	Computes the y-Nearest-Neighbours for the given counterfactuals.
Redundancy	Calculates the number of unnecessary proposed changes.
Success Rate	Measures the fraction of correctly classified counterfactuals with respect to the desired class.
Average Time	The average time counterfactual explanation method needs to generate its result.

Table 2: Description of evaluation metrics implemented in CARLA [42].

To assess the library CARLA, four main factors are closely examined: documentation, dependencies, stability, and usability. The documentation helps the tool's accessibility by providing information about the tool so that users may understand and use it as efficiently as possible. The dependencies of a tool are critical in determining any compatibility difficulties. So that users can run the tool on their machines as intended. The tool's stability will be evaluated in order to identify its dependability and robustness in various circumstances. Finally, the tool's usability is assessed in relation to the existing POC to determine whether the tool or components of the tool can be integrated with the POC and what benefits and drawbacks this would entail.

5.1.1 Documentation

CARLA provides extensive documentation to users to be able to understand the library. A summary page in the documentation provides a brief overview of the library. It includes an installation guide, full component documentation, tutorial notebooks, etc. The documentation is simple to understand and provides a great basis for users. It also includes references to the counterfactual explanation methods implemented in the library. This is required since the tool has been updated with newer methods compared to the version mentioned in the original paper [42].

However, the documentation is not updated as frequently as the changes that can be tracked on the library's github. There is also no version guide available, which is necessary if the library is updated on regular basis. Not all tutorials could be replicated while testing the library. For example, following the benchmarking tutorial notebook, which used CCHVAE method to generate counterfactuals, resulted in internal errors. For testing, CARLA version 0.0.5 was used. The installation instructions available in the documentation and github differs as well. CARLA installation might require the installation of external libraries.

In summary, CARLA offers good and substantial documentation that can be improved with regular updates and the inclusion of a more comprehensive installation guide.

5.1.2 Dependencies

Given the various functions that CARLA provides, the library CARLA has a large number of dependencies. It is dependent on other libraries such as Tensorflow, Pytorch, Sklearn, and others because there are various machine learning models available for the user to train their data on. Users can employ neural networks, linear models, and tree-based algorithms. Although providing customers with a diverse set of models is beneficial, it introduces dependency requirements on other libraries. It is sometimes preferable to have few dependencies to facilitate smooth integration with different environments.

The dependency issue with CARLA is largely related to the installation. CARLA requires users to use Python version 3.7, and any newer version will cause installation issues. This is because the Tensorflow and Pytorch libraries used in the application require that specific version of Python. This may cause compatibility issues because dependant libraries, such as Pytorch, release new versions on a regular basis, and prior versions could be missing maintenance and upgrades, leading in performance issues.

The library was tested on Windows, and no major compatibility difficulties occurred. Running the application on macOS may necessitate the installation of additional external libraries in order to install Pytorch for CARLA.

5.1.3 Stability and Usability

The CARLA library is mostly stable since it includes extensive documentation and robust test suites to assure proper functionality and reliability. However, as previously stated, the library lacks versioning methods. As anybody may contribute to CARLA, the library also adheres to typical quality assurance standards such as code reviews via pull requests and issues on github for tracking new development.

CARLA is a great standalone library for researchers wishing to benchmark counterfactual explanation systems. CARLA's API allows users to use their custom datasets and machine learning models. If the library does not have a counterfactual method, it can be added through github. Using CARLA with another library, on the other hand, can be difficult. Utilising libraries that rely on newer Python versions will cause compatibility issues with CARLA.

5.2 Tool Comparison and Integration Decision

In terms of functionality, CARLA is quite similar to the existing POC. The user selects a dataset, machine learning model, explanation method, and metrics for evaluation. Besides the distance measures, the other evaluation metrics differ from the existing POC. CARLA's output format is also similar to the existing POC in that it displays the output to the user in a tabular style.

Due to the issues raised in the preceding sections about the tool's dependencies, documentation, and stability, it was decided not to integrate any CARLA components with the existing POC. Since it is heavily reliant on other libraries, it is difficult to establish seamless interaction with different environments. Furthermore, CARLA, like the current version of POC, generates output in tabular format, which is considered a shortcoming of the tool as it is harder to interpret. As a result, the existing POC has been expanded for this project by adding an automated module of properties and metrics and presenting the output to the user in a different format.

6 System Design & Implementation

This section introduces the extended pipeline's components. The system design for the pipeline version 2 is provided. The user preference-based system is also discussed, as is the mapping of properties to metrics. The implementation of the pipeline version 2 is also detailed, along with two algorithms.

6.1 System Design

The pipeline version 1 was divided into four major sections: the dataset stage, the ML model stage, the counterfactual explanation methods stage, and the result stage. The objective of the extension was to add an automatic inference of properties module to the prior version, which will map properties and metrics together and can be used to evaluate the explanation approach. Furthermore, the pipeline's entire process was expanded to make it more user-driven. As a result, the end-user is involved in the selection process of datasets, machine learning models, and properties. Making the extended pipeline (pipeline version 2) into a user preference-based system. Table 3 lists the pipeline version 2's built-in features for datasets, ML models, and counterfactual explanation methods. Figure 5 depicts the system design for the pipeline's extended version, which includes automated inference of properties module. This figure is comparable to Figure 4, which depicts the pipeline version 1. However, the user preference-based approach is not evident from Figure 5, and it will be discussed in further detail in the following section.

Categories	Values
Datasets	iris, diabetes, telecom
ML-models	logistic regression, random forest, decision trees, XGBoost
Counterfactual explanation methods	DiCE and Wachter method

Table 3: Built-in Features of the Tool: Datasets, ML-Models and Methods.

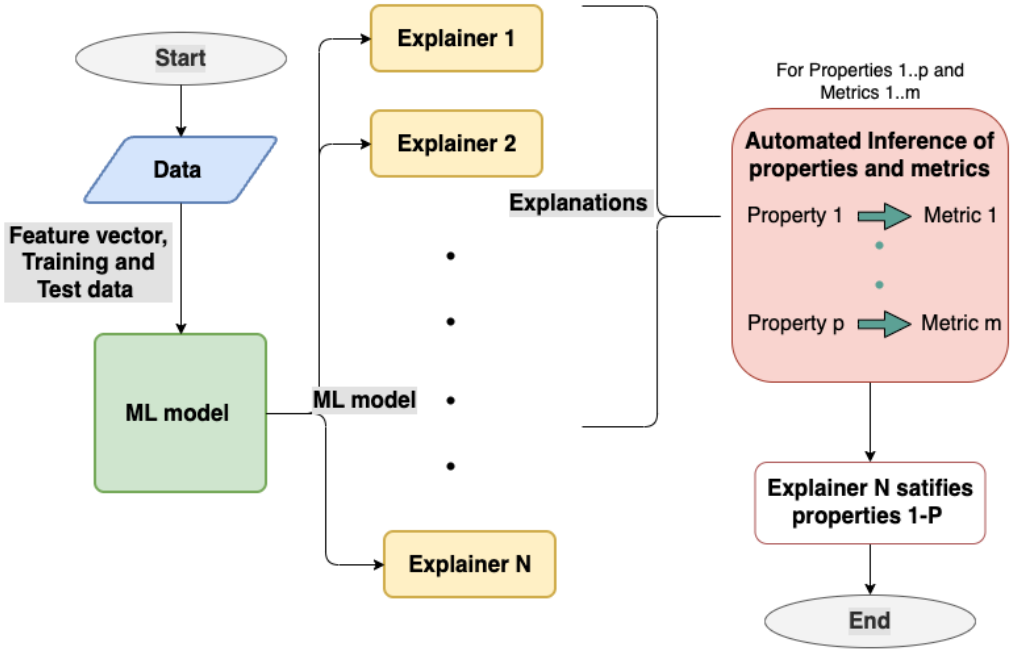

Figure 5 System Design of pipeline version 2.

6.1.1 User Preference-Based System

The user preference-based system implies that the system output is tailored to the user's individual demands by utilising the user preferences across many alternatives and scenarios. In the context of the extended pipeline, this means that the user is requested to choose between a number of datasets, ml-models, and features in order to evaluate and compare the counterfactual explanation. The process of this preference-based system is depicted in Figure 6.

First, the user is requested to select a dataset from the three options indicated in Table 3. After selecting the dataset, the user is prompted to select the original instance class from the dataset's class labels. If the dataset has multi-class classification, the user is then requested to select the desired class. This option is not presented to users for binary classification problems because the opposite of the original instance class is automatically selected. Following that, the user is presented with the dataset's features and requested to select sensitive features. The sensitive features in this scenario are those that are immutable and for which the counterfactual explanation methods should not suggest a change. This input is later utilised to compute a property. The user is subsequently asked to choose a ml-model to train the dataset on, and the implemented counterfactual methods are utilised to generate counterfactuals for a random instance from the original

class. The user is next presented with six desirable properties for counterfactuals that can be used to evaluate the counterfactual explanation methods. The user can select one or more properties. It is worth noting that when multiple properties are selected, a weighted sum is calculated. The weighted sum is computed for metrics results that are mapped to the selected properties. The order in which the user selects the properties determines the weight. To compare methods, the specified properties metric is applied to counterfactuals of one method and the result is saved. The procedure is then repeated for the other method. Once the outcomes for both approaches have been saved, they are compared, and the user is given an output indicating which explanation method best satisfies those properties.

6.1.2 Properties & Metrics Mapping

The mapping of properties chosen to evaluate the counterfactual explanation methods with the metrics is presented in this section. Section 2.2.2 goes into greater depth about these properties, while Section 4 goes into greater detail about the metrics. Table 4 depicts the mapping of properties and metrics.

Propeties	Metrics
Proximity	Feature-wise L1 distance divided by MAD
Validity	Ratio of correctly classified counterfactuals
Sparsity	L2 distance and Hamming distance
Diversity	Cosine Similarity
Actionability	Number of sensitive features violated
Data Manifold Closeness	LOF (Local Outlier Factor)

Table 4: Properties & Metrics Mapping.

6.2 Implementation

In this section, we will discuss what libraries were used for what reasons, how metrics functions were created, and present algorithms for non-trivial functions. The pipeline is written in Python and makes use of widely used libraries such as Numpy [18], Scikit-learn [44], and Pandas [61].

6.2.1 Datasets

There are three datasets that have been added. The Iris dataset [13], provided by scikit-learn, is used. The dataset is made up of three target classes, four features, and 150 samples in total.

The diabetes dataset provided on Kaggle [41] is used. This is a binary classification dataset. It has 9 features and 100000 samples. The dataset is loaded and pre-processed using Pandas. The ”gender” feature column is binary encoded with pre-processing, while the ”smoking_history” column is not used.

The third dataset is a telecom domain dataset that was presented in the paper [53]. This binary classification dataset has 13 features and around 10000 samples.

The datasets are split using scikit-learn's *train_test_split* method, with a test size of 20%.

6.2.2 ML-Models and Counterfactual Explanation Methods

Four machine learning models have been added. The scikit-learn library is used to import logistic regression, random forest, and decision trees. The XGBoost is taken from the XGBoost library [6]. All of the models are built with the default parameters.

There are two counterfactual explanation approaches employed. The DiCE library [39] is imported to implement DiCE method. The Wachter method is implemented with the mlxtend [47] package, and counterfactuals are generated with the *create_counterfactual* method.

6.2.3 Metrics Functions

Most metric functions require an original instance as well as a list of counterfactuals. Other input may be required depending on the need; for example, the model is required for the validity metric. The list of counterfactuals belongs to a single counterfactual method and is used to select the best counterfactual of that method so that it can be compared to the others. For example, if two methods each generate three counterfactuals, the best counterfactual of each method defined by the property metric is compared to each other, also determined by the property metric. The user selects the original instance class, and a random instance is chosen from that class to generate counterfactuals.

The feature-wise L1 distance divided by MAD is used to calculate proximity. The idea behind this is that literature [39] suggests that deviation from the median gives a robust measure of a feature's variability, and therefore dividing by the MAD reflects the relative prevalence of witnessing the feature at a specific value. For validity, a ratio of correctly classified counterfactuals is determined. Algorithm 1 shows the process for calculating

this ratio.

Algorithm 1: Compute Validity ratio

Input : $desired\_class\_label, list\_cfs, model$
Output: $validity\_ratio$

Function `ComputeValidityRatio(`$desired\_class\_label, list\_cfs, model$`)`**:**

$count = 0$

foreach *counterfactual* ***in*** *list_cfs* **do**

$class\_prob = model.predict\_proba(counterfactual)$
$max\_idx = argmax(class\_prob)$

if *desired_class* == *max_idx* **then**
$count \mathrel{+}= 1$
end

end

$validity\_ratio = count/n$

return $validity\_ratio$

Two separate things are measured when it comes to sparsity. The hamming distance is used to calculate how many features are modified, and L2 distance is used to measure the magnitude of the feature change. Cosine similarity is used to assess how different in nature the generated counterfactuals are. The comparison of counterfactual explanation methods is based on the number of distinct counterfactuals produced by the methods. If both methods generate three counterfactuals, cosine similarity is used to determine which method generates the most distinct counterfactual out of the three. Actionability is measured by calculating constraint violations. The sensitive features are chosen by the user, and the generated counterfactuals are examined to see if any of the sensitive aspects are suggested to change by the methods. Each modification in a sensitive feature is counted as a separate violation, and the total number of violations is compared amongst counterfactual explanation approaches. Scikit-learn's LOF is utilised for outlier detection for the data manifold closeness attribute. The *LocalOutlierFactor* estimator is fitted to the data, and the outlier scores are produced based on the local density of each sample. The $fit\_predict$ method is used which provides an array of outlier scores, with 1 indicating an inlier and -1 indicating an outlier. The procedure for detecting outliers is shown in Algorithm 2.

Algorithm 2: Compute Outlier count

Input : $data, list\_cfs$
Output: $count$

Function `ComputeOutlierCount` $(data, list\_cfs)$**:**

$n = len(data)$
$combined\_data = data + list\_cfs$

$lof = LocalOutlierFactor()$
$outlier\_labels = lof.fit\_predict(combined\_data)$

$outlier\_flags = [label == -1$ for label in $outlier\_labels[n:]]$

$count = sum(outlier\_flags)$

return $count$

6.2.4 Weights Calculation

When multiple properties are selected, then a weighted sum is used to calculate the metric's result. Each property is assigned a weight, and the order user chooses the properties determines it. Ordinal preferences determine the weight calculation. Section 4.3 includes more details regarding ordinal preferences. Assuming the index of properties ranges from 0 to n-1, where n is the number of properties selected by the user, then the weightage is calculated as:

$$weights_i = (n - i) / \left[\frac{n(n+1)}{2}\right] \tag{11}$$

7 Results And Discussion

In this section, the outcomes of the experiments are presented and discussed. Experiments are conducted to investigate how various properties can be used to compare counterfactual explanation methods. All of the different implemented datasets and ml-models are utilised for testing in order to acquire a more thorough understanding of the method comparison and demonstrate how user preferences might affect the outcomes.

7.1 Experiments

The experiments are carried out by adding an experiment module to the pipeline. The only difference between this module and the original pipeline is that instead of choosing a random instance from the original class that the user has chosen, all instances from the original class are chosen to run experiments for a particular property. The metric functions described in Table 4 are used for testing different properties. Different datasets are used to conduct tests for different properties. However, not all datasets and ML-models are utilised for evaluating all possible properties.

Validity

The iris dataset is used to evaluate the validity property.

Original class: setosa
Desired class: versicolor

For each instance, DiCE generates three counterfactuals while the Baseline method generates one. The DiCE method generated valid counterfactuals for each instance, indicating that the counterfactual belonged to the versicolor class, whereas the Baseline method was unable to generate a single valid counterfactual for any instance. All counterfactuals generated by the Baseline method belonged to the original class: setosa. The Table 5 displays the outcome of a counterfactual generated and its predicted class using the decision tree method for one instance.

Instance	petal length	petal width	sepal length	sepal width	Predicted Class
Original instance	5.5	3.5	1.3	0.2	Setosa (0)
DiCE 1	5.5	3.5	4.1	0.2	Versicolor (1)
DiCE 2	7.7	3.5	3.8	0.2	Versicolor (1)
DiCE 3	5.5	4.4	3.7	0.2	Versicolor (1)
Baseline Method	5.5	3.6	1.3	0.2	Setosa (0)

Table 5: Feature values for generated counterfactuals.

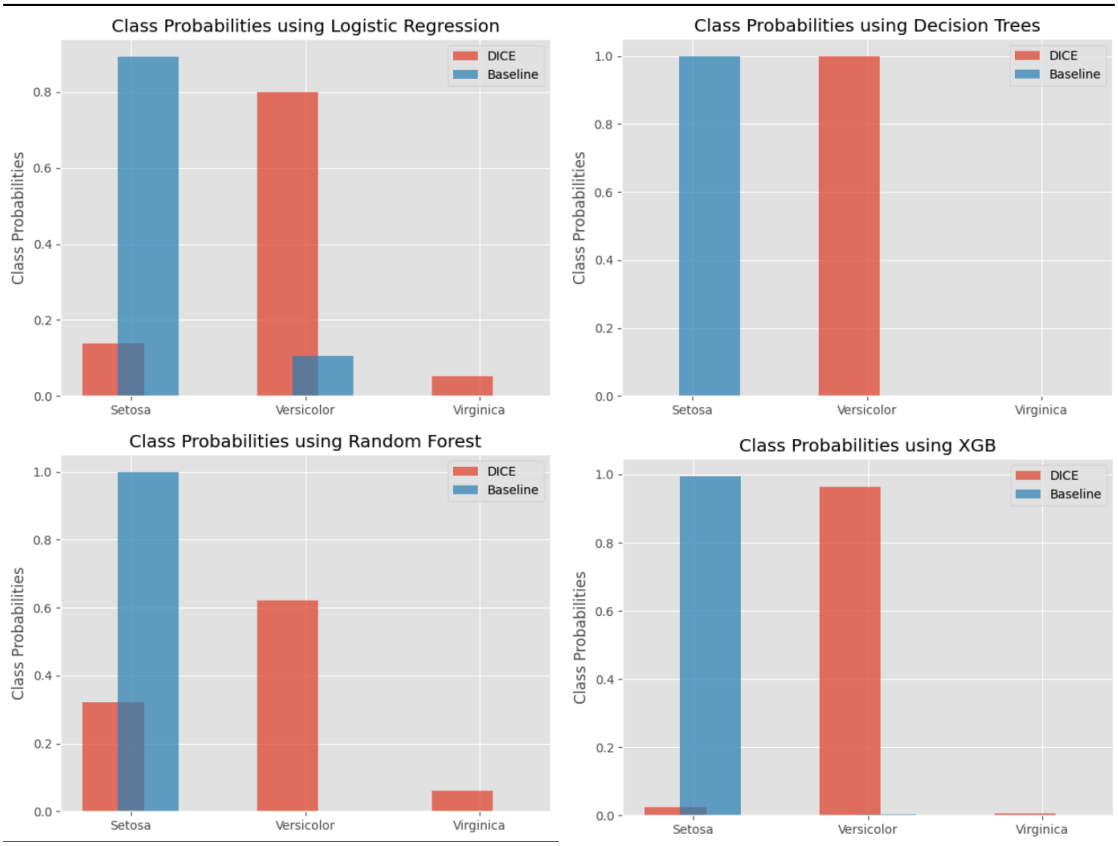

Figure 6 Class probabilities for DiCE & Baseline method from different ML-models.

Figure 6 depicts the class probabilities obtained from several ML-models when applied to counterfactuals generated using the DiCE and Baseline approaches. The mean

of class probabilities is used since counterfactuals for all original class instances were generated and class probabilities for all such counterfactuals were predicted. DiCE produces valid counterfactuals with 100% accuracy whereas Baseline method does not produce a single valid counterfactual.

Proximity & Validity

The diabetes dataset is used to evaluate proximity and validity properties combined. The validity property was selected first to give it a larger weightage in the weightage sum computation.

Original class: 1 (Have diabetes)
Desired class: 0 (No diabetes)

Since the dataset is fairly large , a random sample of 1000 instances from the original class are chosen to run tests on. The DiCE method generates three counterfactuals for each instance, while the Baseline method generates one. The metric associated with the proximity property is calculated for each counterfactual created by DiCE. To compare with the Baseline counterfactual, the DiCE counterfactual with the closest proximity to the original instance is chosen.

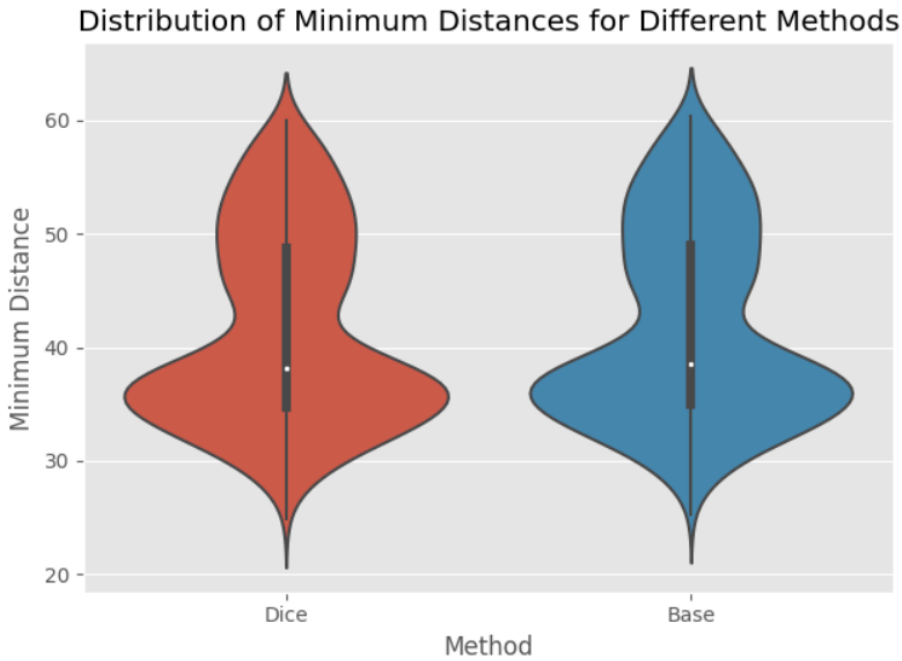

Figure 7 Distribution of minimum distance for DiCE & Baseline methods.

Figure 7 depicts the distribution of minimum distances obtained when the proximity

metric function is employed to analyse various DiCE and Baseline counterfactuals. Logistic Regression when used as the training model for this experiment. The distribution of minimum distances for both methods is quite similar, with DiCE having an average minimum distance of 41.33 and Baseline having 41.64. Since the experiment was designed to examine both proximity and validity, validity was given a greater emphasis. i.e. it was chosen first in order to give it a greater weightage for the weighted sum calculation. Figure 8 presents a graph that shows how many valid counterfactuals were created by each approach when employed with different ML-models. Taking into account the results of logistic regression, where all counterfactuals created by DiCE were valid, i.e. 1000, while Baseline provided 990 valid counterfactuals out of 1000. This finding demonstrates that DiCE generated better counterfactuals that fulfilled the two criteria.

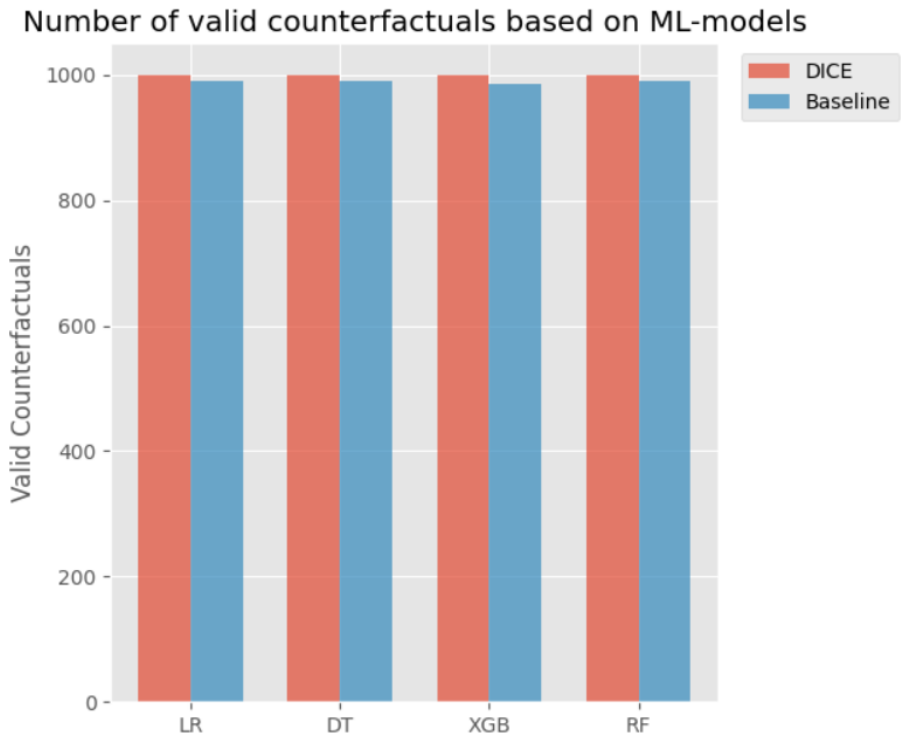

Figure 8 Number of valid counterfactuals generated by method based on different ML-models.

Sparsity & Actionability

The telecom and diabetes datasets are used to test the properties sparsity and actionability. The actionability property was selected first to give it a larger weightage in the weightage sum computation.

Telecom dataset
Original class: 1 (Violation)
Desired class: 0 (No violation)

Diabetes dataset
Original class: 1 (Have diabetes)
Desired class: 0 (No diabetes)

Dataset	Total features	Sensitive features	Constraint Violations DiCE	Features change DiCE	Constraint Violations Baseline	Features change Baseline
Diabetes	9	2	0	1.2	1.59	4.9
Telecom	13	3	0	1.2	2.8	11.8

Table 6: Average Features change suggested & Constraint violations by DiCE and Baseline methods.

To assess Actionability, it was decided to analyse how many sensitive attributes specified by the user are violated by each counterfactual method. Sparsity is assessed by counting the number of features proposed to modify by each generated counterfactual. For DiCE three counterfactuals were generated for each instance and for baseline 1. Table 6 displays the results of an experiment conducted on two datasets: diabetes and telecom. For diabetes, two sensitive features were chosen: age and gender, while for telecom, three random features were chosen as sensitive. This was done due to lack of domain knowledge.

When tested on these properties, DiCE outperformed the baseline method by a wide margin. Table 6 displays the average results for diabetes and telecom datasets which was conducted on 1000 randomly selected instances. This was done to save time, as both datasets were rather large and the counterfactual generation is a time-consuming process. The DiCE method did not violate any constraints in either dataset, but the baseline method violated on average 1.59 features out of 2 sensitive features in the diabetes dataset and 2.8 out of 3 sensitive features in the telecom dataset. For both

datasets, baseline method suggested significantly more feature changes than the DiCE method.

Data manifold closeness

The iris dataset is used to evaluate the Data manifold closeness property for DiCE and Baseline methods.

Original class: 0 (setosa)
Desired class: 1 (versicolor)

The counterfactuals were generated for ten instances of the original class, 0 (setosa). For each instance, both methods produced one counterfactual. For each counterfactual of both approaches, the metric function for data manifold closeness property was run using all four ML-models.

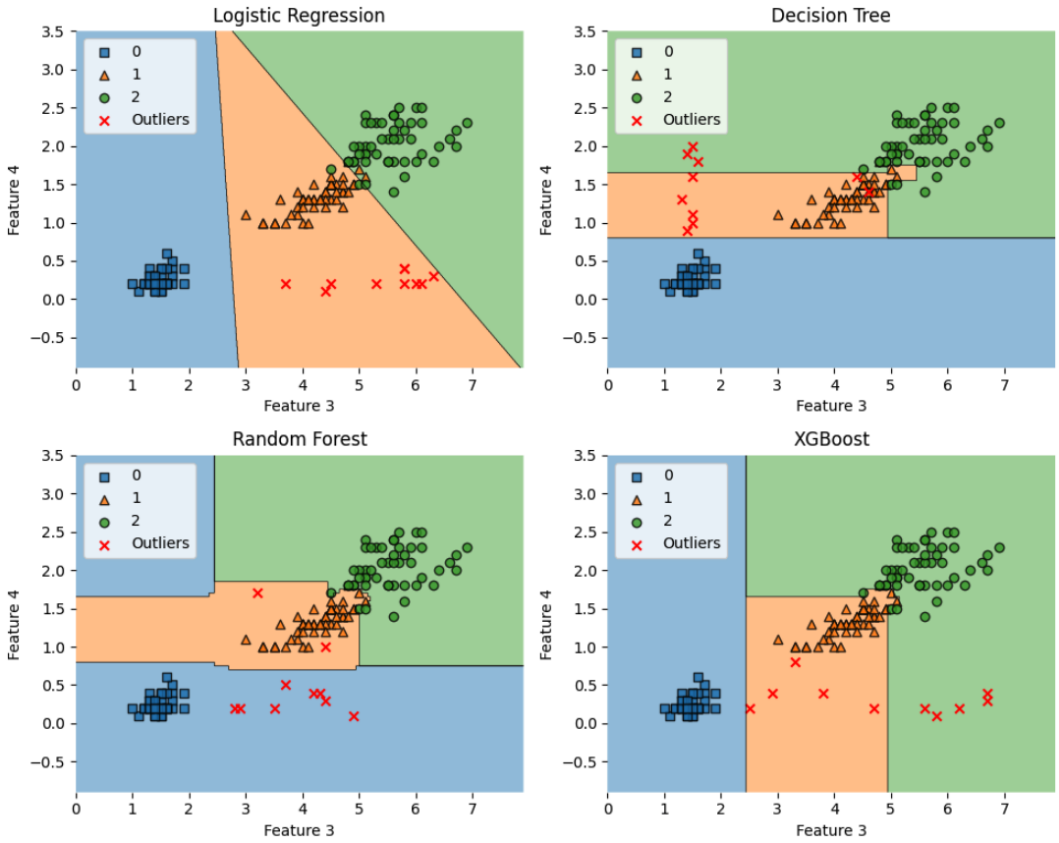

Figure 9 Scatter plot illustrating the decision boundaries and outliers obtained using different ML-methods for DiCE.

Figures 9 and 10 depict the end outcome of the experiment. Figure 9 depicts the con-

clusion of how DiCE-generated counterfactuals were labeled as outliers or not. It is illustrated that each DiCE-generated counterfactual was categorised as an outlier by all ML-models. For logistic regression, all counterfactuals were categorised in the desired class, 1 (versicolor), and can be considered outliers when compared to class 1 (versicolor) datapoints. However, the results for the other three Ml-models are inconsistent. Some counterfactuals are invalid, i.e. they are not categorised correctly, and some counterfactuals are false positive outliers. In Figure 9, the Decision Tree plot shows that two counterfactuals are incorrectly labeled as outliers because they are located in dense clusters of datapoints.

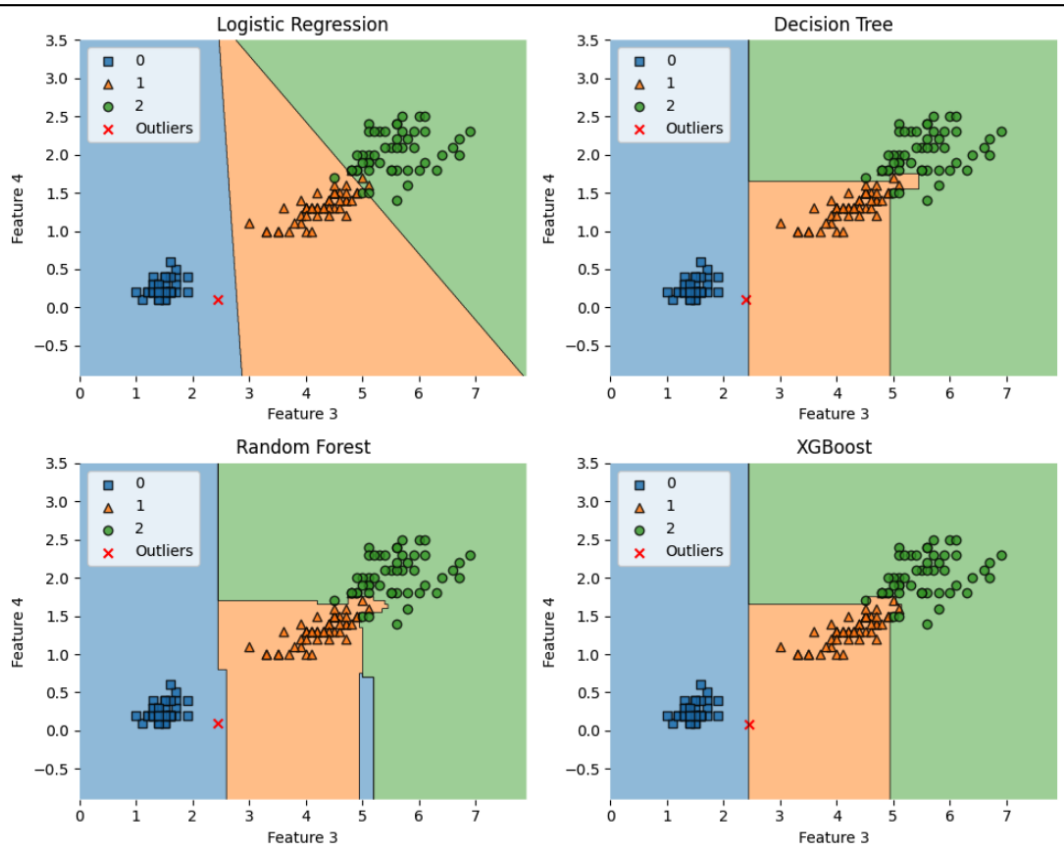

Figure 10 Scatter plot illustrating the decision boundaries and outliers obtained using different ML-methods for Baseline.

Figure 10 shows that when using the Baseline method, only one counterfactual has been classified as an outlier for each ML-model. It is also worth noting that the same counterfactual is identified as an outlier in all Ml-models. The outlier is also an invalid counterfactual for all ML-models since it does not belong to the desired class.

7.2 Discussion

One of the primary objectives of this thesis was to develop a properties metric mapping module that could be used to better compare the implemented counterfactual explanation techniques. The experimentation's aim was to test this module in numerous ways and determine whether a useful comparison between different techniques could be accomplished.

The validity property was examined twice, once by itself and once combined with the proximity property. This is a crucial property for counterfactuals because if the generated counterfactual does not belong to the intended class, the other counterfactual properties such as proximity, sparsity, and so on become insignificant. As mentioned in Section 2.2.2 a higher validity is preferred. The results of the first experiment are shown in Figure 6, where DiCE outperformed the baseline method for generating valid counterfactuals. The baseline method was unable to generate a single valid counterfactual for any instance. The reasoning for this can be the implementation used for the baseline method is incapable of handling multiple classes, as the iris dataset was used for this experiment which is a multiclass classification problem. This was determined after reviewing the findings of the second experiment, which are depicted in Figure 8. The diabetes dataset was used for this experiment which is a binary classification problem. DiCE produced more valid counterfactuals than the baseline method for this experiment as well but only by a small margin. The weighted sum was used to determine whether counterfactual methods achieved both properties in the second experiment, where proximity was also considered in conjunction with validity. The use of a weighted sum proved to be an efficient and comprehensive approach, yielding a well-represented outcome that reflects the collective insights from the chosen metrics. DiCE had a higher weighted sum because it was slightly better than the baseline technique in producing both valid and closer counterfactuals.

The properties of sparsity and actionability were examined collectively, and the weighted sum of the metrics results was calculated. As seen in Table 4, the sparsity property is associated with two separate metrics. It can be used to determine the amount of the suggested modification as well as the number of suggested feature changes. It was determined that testing the number of feature changes in combination with actionability, which determines the number of constraint violations, was sufficient for this purpose. For this experiment, two separate binary classification datasets were utilised. Table 6 shows that DiCE performed much better than the baseline method for both properties independently. In this instance, user preferences for assigning weight to different properties is inconsequential considering any ordering of properties will have DiCE satisfying both properties better than the baseline method for this particular setup.

The data manifold closeness property was tested using the iris dataset. The property's

outlier detection metric was utilised to identify whether the created counterfactuals were outliers or not. All four ML-models were utilised for both counterfactual approaches, and the associated findings are displayed in Figures 9 and 10. Baseline fared significantly better than DiCE in that it produced just one counterfactual, which was considered to be an outlier. Figure 10 shows that the counterfactual was appropriately categorised as an outlier since it was distant from the desired class versicolor cluster and belonged to the original class cluster. Figure 9 reveals that numerous DiCE counterfactuals are incorrectly labelled as outliers since they are near or in the cluster of the desired class (1). It is crucial to note that in Figures 9 and 10, the scatter plots with decision boundaries use only features: petal length (cm) and petal width (cm) from the iris dataset to simplify the plot. These features are labelled as feature 3 and 4 respectively in Figures 9 and 10. By concentrating primarily on these two features, there is a risk of oversimplifying the underlying categorisation process and perhaps neglecting vital information conveyed by the other two features. Consequently, the decision boundary depicted in the scatter plot may not precisely represent the real decision boundary when all four features are considered. Feature 3 and 4 were chosen because counterfactuals showed the most variation in these two attributes.

The diversity property was not included in the experimentation module. It was determined that comparing DiCE with the baseline method for diversity property was not fair since the baseline method is unable to generate different counterfactuals for the same instance. In every scenario, DiCE will outperform the baseline method in terms of diversity. Other counterfactuals methods capable of generating multiple counterfactuals for the same instance need to be implemented in the pipeline to test the diversity property.

The user preferences were a crucial component of the new pipeline module. Few random user preferences scenarios were built for testing. However, user evaluation on the new module can be beneficial. The user assessment was considered challenging due to time constraints as it would require to organise many evaluation sessions and work on the interface of the pipeline for presenting the results to the users. These were not part of the project's scope and may have caused delays.

8 Conclusions

To summarise, this research project was successful in extending the existing POC by improving its modularity by adding a new mapping module for properties and metrics. This addition is an important step toward offering a standardised benchmarking tool for field researchers. The substantial literature research (described in Section 2) done as part of this project was critical in the development of the new module. A thorough examination of existing literature yielded a thorough grasp of the many properties and metrics associated with counterfactual explanation approaches. This aided in determining the most appropriate properties and evaluation metrics, which were then included into the new module. Furthermore, the insights gained from the literature research contributed in defining the mapping for the properties with the metrics. The result stage indicated in Section 3.4 was redefined to incorporate the new mapping module of properties and metrics after examining various perspectives and approaches from the literature.

The user-preference-based method, which provided a dynamic and individualised way of assessing counterfactual explanations, was a key component of the new module. It empowered users by allowing them to choose a specific dataset, ml-models, and properties on which they would evaluate counterfactuals, enabling them to personalise the analysis to their own needs. The user preference system's increased flexibility encourages transparency and inclusiveness, since diverse target audiences such as researchers, stakeholders, and domain experts can utilise this tool based on their requirements.

8.1 Challenges & Limitations

The mapping of properties to suitable evaluation metrics for counterfactual explanations presented a significant challenge due to several factors. Firstly, counterfactual explanation comprises a wide range of desired features, each of which provides a distinct aspect of the explanation. This necessitates a variety of evaluation methodologies. Second, the relation between properties and metrics is complex. It is not trivial since it is not necessarily a one-to-one mapping. For example, properties such as proximity have well-defined measurements, and any of these distinct distance metrics may be employed to measure based on the need. In contrast, there are no universally accepted measuring metrics for properties such as validity and diversity.

A further challenge was establishing the user's role. Considering this is a user-preference based system, it was necessary to analyse how various audiences would approach the solution. This entailed estimating how much domain knowledge the user would need to use this application. As the sensitive features were chosen by the user, they needed to some knowledge about the datasets. It was also difficult to identify how the output should be presented to the user so that it could be easily interpreted.

The following are some of the tool's limitations. As previously stated, one of the constraints is that users must have prior knowledge of datasets in order select the sensitive features. In terms of comparing the counterfactual methods, the diversity property cannot be evaluated fairly. Since the baseline method is incapable of generating several counterfactuals for the same instance. When a user chooses the original class, a random instance from that class is picked to produce counterfactuals for. It would be preferable to allow the user to manually specify an instance for which counterfactuals should be created.

8.2 Future Work

There are various routes to pursue in order to scale and improve the POC's capabilities. To begin, the POC can be expanded by including a broader range of datasets. Currently, only tabular datasets are implemented, however image datasets can be added in the future. More complicated tabular datasets with more features and a wider range of feature types can also be included. Furthermore, other machine learning models could be incorporated to handle varied kinds of datasets, such as convolutional neural networks for image datasets. The POC can further be expanded by including more properties for counterfactual explanation methods as well as other metrics for evaluating them.

The tool is currently terminal-based. To make the tool more user-friendly and easier to use, a GUI (Graphical User Interface) can be included. Once the tool has a GUI, user evaluations can be used to enhance the output format, i.e. what information is helpful to the users and what is not.

References

[1] C. Agarwal, S. Krishna, E. Saxena, M. Pawelczyk, N. Johnson, I. Puri, M. Zitnik, and H. Lakkaraju, "OpenXAI: Towards a Transparent Evaluation of Model Explanations," 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2206.11104

[2] J. Antorán, U. Bhatt, T. Adel, A. Weller, and J. M. Hernández-Lobato, "Getting a clue: A method for explaining uncertainty estimates," 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2006.06848

[3] C. B. Azodi, J. Tang, and S.-H. Shiu, "Opening the Black Box: Interpretable Machine Learning for Geneticists," *Trends in Genetics*, vol. 36, no. 6, pp. 442–455, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016895252030069X

[4] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Information Fusion*, vol. 58, pp. 82–115, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253519308103

[5] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying Density-Based Local Outliers," *SIGMOD Rec.*, vol. 29, no. 2, p. 93–104, may 2000. [Online]. Available: https://doi.org/10.1145/335191.335388

[6] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. ACM, 2016, pp. 785–794. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785

[7] A. Dhurandhar, P.-Y. Chen, R. Luss, C.-C. Tu, P. Ting, K. Shanmugam, and P. Das, "Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives," *Advances in neural information processing systems*, vol. 31, 2018.

[8] A. Dhurandhar, P.-Y. Chen, R. Luss, C.-C. Tu, P. Ting, K. Shanmugam, and P. Das, "Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1802.07623

[9] F. Doshi-Velez and B. Kim, "Towards A Rigorous Science of Interpretable

Machine Learning," 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1702.08608

[10] M. Dreyer, F. Pahde, C. J. Anders, W. Samek, and S. Lapuschkin, "From Hope to Safety: Unlearning Biases of Deep Models by Enforcing the Right Reasons in Latent Space," 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2308.09437

[11] P. Fishburn, "Utility and subjective probability: Contemporary theories," in *International Encyclopedia of the Social & Behavioral Sciences*, N. J. Smelser and P. B. Baltes, Eds. Oxford: Pergamon, 2001, pp. 16 113–16 121. [Online]. Available: https://doi.org/10.1016/B0-08-043076-7/00638-0

[12] P. C. Fishburn, "Utility Theory," *Management Science*, vol. 14, no. 5, pp. 335–378, 1968.

[13] R. A. Fisher, "The Use of Multiple Measurements in Taxonomic Problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[14] J. González-Pachón, L. Diaz-Balteiro, and C. Romero, "How to combine inconsistent ordinal and cardinal preferences: A satisficing modelling approach," *Computers & Industrial Engineering*, vol. 67, pp. 168–172, 2014. [Online]. Available: https://doi.org/10.1016/j.cie.2013.11.008

[15] R. Guidotti, "Counterfactual explanations and how to find them: literature review and benchmarking," *Data Mining and Knowledge Discovery (2022)*, 22. [Online]. Available: https://doi.org/10.1007/s10618-022-00831-6

[16] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, D. Pedreschi, and F. Giannotti, "A Survey Of Methods For Explaining Black Box Models," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1802.01933

[17] D. Gunning and D. Aha, "DARPA's Explainable Artificial Intelligence (XAI) Program," *AI Magazine*, vol. 40, no. 2, pp. 44–58, Jun. 2019. [Online]. Available: https://doi.org/10.1609/aimag.v40i2.2850

[18] C. R. Harris, K. J. Millman, S. J. der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[19] M. R. Islam, M. U. Ahmed, S. Barua, and S. Begum, "A Systematic Review of Explainable Artificial Intelligence in Terms of Different Application Domains and Tasks," *Applied Sciences*, vol. 12, no. 3, p. 1353, 2022. [Online]. Available: https://doi.org/10.3390/app12031353

[20] C. Jatoth, N. E., M. A.V.R., and S. R. Annaluri, "Effective monitoring and prediction of Parkinson disease in Smart Cities using intelligent health care system," *Microprocessors and Microsystems*, vol. 92, p. 104547, 2022. [Online]. Available: https://doi.org/10.1016/j.micpro.2022.104547

[21] S. Joshi, O. Koyejo, W. Vijitbenjaronk, B. Kim, and J. Ghosh, "Towards realistic individual recourse and actionable explanations in black-box decision making systems," 2019.

[22] K. Kanamori, T. Takagi, K. Kobayashi, and H. Arimura, "DACE: Distribution-Aware Counterfactual Explanation by Mixed-Integer Linear Optimization," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, C. Bessiere, Ed. International Joint Conferences on Artificial Intelligence Organization, 7 2020, pp. 2855–2862. [Online]. Available: https://doi.org/10.24963/ijcai.2020/395

[23] A.-H. Karimi, B. Schölkopf, and I. Valera, "Algorithmic Recourse: from Counterfactual Explanations to Interventions," 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2002.06278

[24] W. Kenton, "What is a black box model? definition, uses, and examples," 2023. [Online]. Available: https://www.investopedia.com/terms/b/blackbox.asp

[25] V. Kotu and B. Deshpande, "Chapter 13 - Anomaly Detection," in *Data Science (Second Edition*, V. Kotu and B. Deshpande, Eds. Morgan Kaufmann, 2019, pp. 447–465. [Online]. Available: https://doi.org/10.1016/B978-0-12-814761-0.00013-7

[26] S. Krishna, T. Han, A. Gu, J. Pombra, S. Jabbari, S. Wu, and H. Lakkaraju, "The Disagreement Problem in Explainable Machine Learning: A Practitioner's Perspective," 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2202.01602

[27] T. Laugel, M.-J. Lesot, C. Marsala, X. Renard, and M. Detyniecki, "Inverse Classification for Comparison-based Interpretability in Machine Learning," 2017.

[28] D. K. Lewis, *Counterfactuals*. Malden, Mass.: Blackwell, 1973.

[29] A. Lindholm, N. Wahlström, F. Lindsten, and T. B. Schön, *Machine Learning -*

A First Course for Engineers and Scientists. Cambridge University Press, 2022. [Online]. Available: https://smlbook.org

[30] Z. C. Lipton, "The Mythos of Model Interpretability," 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1606.03490

[31] A. V. Looveren and J. Klaise, "Interpretable Counterfactual Explanations Guided by Prototypes," 2020. [Online]. Available: https://doi.org/10.48550/arXiv.1907.02584

[32] O. Loyola-González, "Black-Box vs. White-Box: Understanding Their Advantages and Weaknesses From a Practical Point of View," *IEEE Access*, vol. 7, pp. 154 096–154 113, 2019. [Online]. Available: https://doi.org/10.1109/ACCESS.2019.2949286

[33] A. Lucic, H. Oosterhuis, H. Haned, and M. de Rijke, "FOCUS: Flexible Optimizable Counterfactual Explanations for Tree Ensembles," 2021. [Online]. Available: https://doi.org/10.48550/arXiv.1911.12199

[34] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[35] D. Mahajan, C. Tan, and A. Sharma, "Preserving Causal Constraints in Counterfactual Explanations for Machine Learning Classifiers," 2020. [Online]. Available: https://doi.org/10.48550/arXiv.1912.03277

[36] S. Mahmoud, E. Billing, H. Svensson, and S. Thill, "Where to from here? on the future development of autonomous vehicles from a cognitive systems perspective," *Cognitive Systems Research*, vol. 76, pp. 63–77, 2022. [Online]. Available: https://doi.org/10.1016/j.cogsys.2022.09.005

[37] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial Intelligence*, vol. 267, pp. 1–38, 2019. [Online]. Available: https://doi.org/10.1016/j.artint.2018.07.007

[38] C. Molnar, *Interpretable Machine Learning*, 2nd ed., 2022. [Online]. Available: https://christophm.github.io/interpretable-ml-book

[39] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," in *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. ACM, jan 2020. [Online]. Available: https://doi.org/10.48550/arXiv.1905.07697

[40] A. Mujumdar, K. Cyras, S. Singh, and A. Vulgarakis, "Trustworthy AI: explainability, safety and verifiability," 2020. [Online]. Available: https://www.ericsson.com/en/blog/2020/12/trustworthy-ai

[41] M. Mustafa, "Diabetes prediction dataset," 2023. [Online]. Available: https://www.kaggle.com/datasets/iammustafatz/diabetes-prediction-dataset

[42] M. Pawelczyk, S. Bielawski, J. van den Heuvel, T. Richter, and G. Kasneci, "CARLA: A Python Library to Benchmark Algorithmic Recourse and Counterfactual Explanation Algorithms," 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2108.00783

[43] M. Pawelczyk, K. Broelemann, and G. Kasneci, "Learning Model-Agnostic Counterfactual Explanations for Tabular Data," in *Proceedings of The Web Conference 2020*. ACM, 2020. [Online]. Available: https://doi.org/10.1145/3366423.3380087

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

[45] R. Poyiadzi, K. Sokol, R. Santos-Rodriguez, T. D. Bie, and P. Flach, "FACE: Feasible and Actionable Counterfactual Explanations," in *Proceedings of the AAAI/ACM Conference on AI,Ethics, and Society*. ACM, feb 2020. [Online]. Available: https://doi.org/10.48550/arXiv.1909.09369

[46] A. Preece, D. Harborne, D. Braines, R. Tomsett, and S. Chakraborty, "Stakeholders in Explainable AI," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1810.00184

[47] S. Raschka, "MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack," *The Journal of Open Source Software*, vol. 3, no. 24, Apr. 2018. [Online]. Available: https://joss.theoj.org/papers/10.21105/joss.00638

[48] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why Should I Trust You?": Explaining the Predictions of Any Classifier," 2016. [Online]. Available: https://doi.org/10.48550/arXiv.1602.04938

[49] M. T. Ribeiro, S. Singh, and C. Guestrin, "Achors: High-Precision Model-Agnostic Explanations," *AAAI*, vol. 30, no. 1, 2018. [Online]. Available: https://doi.org/10.1609/aaai.v32i1.11491

[50] D. Rindskopf and M. Shiyko, "Measures of Dispersion, Skewness and Kurtosis," in *International Encyclopedia of Education (Third Edition)*. Oxford: Elsevier, 2010, pp. 267–273. [Online]. Available: https://doi.org/10.1016/B978-0-08-044894-7.01344-0

[51] H. A. Simon, "On the definition of the causal relation," *The Journal of Philosophy*, vol. 49, no. 16, pp. 517–528, 1952.

[52] V. Singh, "Explainable AI Metrics and Properties for Evaluation and Analysis of Counterfactual Explanations," 2021.

[53] A. Terra, R. Inam, S. Baskaran, P. Batista, I. Burdick, and E. Fersman, "Explainability methods for identifying root-cause of sla violation prediction in 5g network," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–7.

[54] G. Tolomei, F. Silvestri, A. Haines, and M. Lalmas, "Interpretable Predictions of Tree-based Ensembles via Actionable Feature Tweaking," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017. [Online]. Available: https://doi.org/10.1145/3097983.3098039

[55] S. Upadhyay, S. Joshi, and H. Lakkaraju, "Towards Robust and Reliable Algorithmic Recourse," 2021.

[56] B. Ustun, A. Spangher, and Y. Liu, "Actionable Recourse in Linear Classification," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*. ACM, 2019. [Online]. Available: https://doi.org/10.1145/3287560.3287566

[57] S. Verma, V. Boonsanong, M. Hoang, K. E. Hines, J. P. Dickerson, and C. Shah, "Counterfactual Explanations and Algorithmic Recourses for Machine Learning: A Review," 2022. [Online]. Available: https://doi.org/10.48550/arXiv.2010.10596

[58] G. Vilone and L. Longo, "Classification of Explainable Artificial Intelligence Methods through Their Output Formats," *Machine Learning and Knowledge Extraction*, vol. 3, no. 3, pp. 615–661, 2021. [Online]. Available: https://doi.org/10.3390/make3030032

[59] J. von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[60] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1711.00399

[61] Wes McKinney, "Data Structures for Statistical Computing in Python," in *Proceedings of the 9th Python in Science Conference*, Stéfan van der Walt and Jarrod Millman, Eds., 2010, pp. 56 – 61.

[62] L. Yang and R. Jin, "Distance Metric Learning: A Comprehensive Survey," *Michigan State Universiy*, vol. 2, no. 2, p. 4, 2006.

[63] L. Zou, "Chapter 3 - Metric-based meta-learning approaches," in *Meta-Learning*, L. Zou, Ed. Academic Press, 2023, pp. 39–59. [Online]. Available: https://doi.org/10.1016/B978-0-323-89931-4.00003-1