Explaining Decisions of Black-box Models using Association Rules

by

Mohammad Hossein Motallebi Shabestari

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Present-day advancements in AI, amongst other things, have often been regarding improving the accuracy of classification models. One lagging aspect, however, is justifying the decisions made by those models. Recently, AI researchers are paying more attention to fill this gap, leading to the introduction of the new field of eXplainable AI (XAI). Model-independent explanations are one class of explanation methods in XAI that aim at addressing the mentioned problem using techniques that have no access to the internals of a learned model.

In this work, we introduce BARBE, a model-independent method that explains the decisions of any black-box classifier for tabular datasets with high precision. Moreover, the black-box classifier is not required to provide any probability score to take advantage of BARBE. Furthermore, BARBE presents explanations in two alternative forms: 1) the importance score for salient features, which many methods also benefit from; 2) construction of rules, which distinguishes BARBE from other methods. Rules are regarded as a better way to provide explanations as they align better with human intuition. Furthermore, BARBE exploits association rules, a special kind of rule that takes into consideration the associations between features, helping users comprehend different underlying causes of a decision.

We also introduce BARBiE, an extension to BARBE that provides interactive explanations. This framework allows users to alter the features of an instance for which the prediction is being explained, and observe how their

modifications affect the explanation and the class label.

Preface

Part of Section 2.1.2 is from [4], a publication where the author was a major contributor. The rest of this manuscript is the original work of the author and is being considered for additional publications.

To my family

For supporting me through every step of my life.

Two there are who are never satisfied – the lover of the world and the lover of knowledge.

– Rumi

Acknowledgements

First and foremost, I would like to express my sincere gratitude to professor Osmar Zaiane. This work would not have been completed without him. The interactions I had with Osmar during my time at the University of Alberta, and the priceless things I learned from him, will always inspire me for the rest of my career.

Additionally, my gratitude extends to all other members of the Amii xAI Lab for the insightful discussions we had. Randy, Mi-Young, Juliano, Housam, Nawshad, Shahin, Talat, and Sheila, thank you!

I would like to thank all my friends in Edmonton who made my time in this beautiful city joyful. Also, I want to pay my respects to my friends Nasim, Pouneh, and Arash who we lost in the tragic airplane crash in January 2020. They will always be missed.

Last but not least, I would like to express my honest appreciation to my family. My mother and father who have always, especially during difficult times, supported me with their unconditional love. It is incomplete if I do not thank my brother, who has always been the first person I have reached out for assistance. He has helped me in different aspects of my life, and for that, I am grateful.

Contents

List of Tables

List of Figures

AI Artificial Intelligence

CNN Convolutional Neural Network

BARBE Black-box Association Rule-Based Explanations

BARBiE Black-box Association Rule-Based Interactive Explanations

CAR Classification Association Rule

CBA Classification Based on Associations

CMAR Classification based on Multiple Association Rules

CPAR Classification based on Predictive Association Rules

CRF Conditional Random Field

DARPA Defense Advanced Research Projects Agency

DNN Deep Neural Networks

EHR Electronic Health Record

EU European Union

ES Expert System

FOIL First Order Inductive Learner

GDPR General Data Protection Regulation

HMM Hidden Markov Model

IR Information Retrieval

KB Knowledge Base

LHS Left Hand Side

LSTM Long Short-term Memory

ML Machine Learning

NER Named-Entity Recognition

NLP Natural Language Processing

NMT Neural Machine Translation

POS Part-Of-Speech

RF Random Forest

RNN Recurrent Neural Network

RBO Rank-Biased Overlap

RHS Right Hand Side

RIPPER Repeated Incremental Pruning to Produce Error Reduction

SVM Support Vector Machine

VQA Visual Question Answering

XAI eXplainable Artificial Intelligence

Chapter 1

Introduction

1.1 Motivation

Recently, more and more companies are benefiting from Artificial Intelligence (AI) in their products. As a result, societies are impacted more by intelligent machines than ever before. The complexity of tasks involving AI varies from simple ones like route-finding to more sophisticated ones such as self-driving cars: Google Maps employs AI to suggest the fastest route to its users, while Tesla Autopilot is replacing humans as the drivers of vehicles. Such advancements, however, started decades ago with the introduction of Expert Systems (ES), the first applications of AI in real-life problems, after the first AI winter, a period of reduced interest in AI and funding cuts. Over the years, more complex intelligent systems were introduced, which differed from ES systems in various aspects.

1.1.1 Expert Systems

One of the early applications of AI that has been employed both in the research community and in commercial settings is Expert Systems. Expert Systems were the first applications of AI in real-life problems at a large scale, where they benefited corporations and helped them save money. R1 [43], for example, helped Digital Equipment Corporations save an estimated $40 million a year [62].

Liao [39] defines Expert Systems as a decision-making software package that performs like humans, if not better, in a narrow and specific problem

area. Any Expert System has two main components: domain knowledge and an inference engine. The domain knowledge is comprised of some general knowledge and facts that are provided by the system users. The inference engine uses a method to make a deduction given the facts and the knowledge base. An Expert System, once provided with the facts, uses them together with its knowledge base to make inferences, acts as a consultant in that specific field, and if needed, can explain the logic behind its advice.

The first Expert System, introduced in 1965, was Dendral [23]. Its purpose was to help chemists identify unknown organic molecules.

Another Expert System was GATES [8], which was an ES that assigned airport gates to arriving or departing flights at JFK Airport in New York City. It used a combination of permissive and conflict rules to assign a gate to a flight.

Buchanan [9] defines Expert Systems along four dimensions: 1) AI methodology: these systems are based on AI and should reason based on some information; 2) High Performance: systems should outperform humans or at least be as good as humans in the task; 3) Flexibility: systems are not algorithms and have a higher tolerance to changes in both design and run-time; 4) Understandability: **they should be able to explain the rationale behind reasoning as human experts can**.

Early versions of ES met all four of these criteria. For example, GATES used a constraint satisfaction approach to find the best gate for each flight. It was also capable of finding the gate for each flight very quickly and as correctly as human experts could. Users were also able to modify it according to changes in the weekly flight schedules. Moreover, users could see the rules used to provide a suggestion for each flight. As can be seen, GATES expert system satisfied all the requirements of an ES defined by Buchanan [9].

1.1.2 Deep Learning Era

While intelligent systems such as Dendral or GATES were great assistants to their users, industry and academia introduced some different AI systems over the years that could handle more sophisticated tasks such as fraud de-

tection [22], or were capable of outperforming humans in tasks such as games like Go [66] or poker [51].

These newer systems process more data and also do more computation than earlier systems, thus outperforming them while also covering more domains.

The knowledge provided to these systems was no longer in the form of a list of rules but comprised of observations or examples with their desired output that the system could use to learn a predictive model. Deep Neural Networks (DNNs) is an example of such sophisticated improvements that, in recent years, have been helping users solve more complex problems compared to earlier methods used in systems such as Dendral.

However, despite being able to tackle complex problems even beyond human capabilities, the new approaches have some shortcomings in other aspects.

1.1.3 A Problem with Recent Systems

One of the main drawbacks in modern systems is the fact that the sophisticated system's underlying decision-making process is not evident to the users. In essence, these systems receive some data as input (e.g., images) and in addition to the class label, they only provide, at most, some probabilities associated with each class as output. Such systems cannot explain their decisions by any means to their users. When a judge uses a sophisticated model for bail decisions, they will not be able to understand the reasoning behind the system's suggestion, and as we can expect, this has caused problems such as the release of dangerous criminals [70]. As another example, when a model is used to recommend a cancer treatment method, neither the patient dealing with cancer nor the doctor would trust the system's suggested treatment unless they understood on what rationale that suggestion was based. This means, unlike Dendral, where the system could explain the decision made by providing some guiding rules used to come up with the final result, many current systems are not capable of providing such information at all.

Black-box model is the term researchers use to refer to such complex models to emphasise this vital shortcoming. Consequently, many users choose to stick with more straightforward, yet more understandable models rather than move

to more accurate but more complicated systems. This is because they can explain the justification behind a decision using the simpler models, as that explanation plays a vital role in their tasks.

Furthermore, laws such as The EU General Data Protection Regulation (GDPR) [24] have come into effect since 2018. GDPR requires explanations for some algorithmic decisions. Finally, this need also complies with the main criteria that were set for Expert systems, one of the early applications of AI.

1.2 Explaining Black-box Systems

As mentioned in the previous section, not only is providing explanations beneficial for the users (For example, in cases such as why a computer made a specific move in the game of Go so players could learn new moves), but it is legally mandatory in many cases because of GDPR, or necessary as in medical domains.

1.2.1 Can we get explanation from black-box models?

To resolve this shortcoming, scientists have recently expanded their research to find ways to provide explanations (or insights) into how models "reason" to help humans more confidently accept the decision of a system. These explanations are usually built by modifying and complementing an existing architecture. They can vary from an image's heat-map (showing the effect of each pixel in the system's final decision using gradients of different layers of a DNN [64]) to providing a sentence describing the content of an image [30].

Despite being an excellent tool for explainability, a big issue such "model-dependent" explainers have is that these systems are only able to explain a specific architecture; if one slightly modifies the model's architecture, or decides to switch to another model for a task, they need a new explainer. Furthermore, many existing models do not provide explanations off-the-shelf. Given that state-of-the-art models change on a yearly -if not monthly- basis, we need different methods that can fill the existing gap and provide explanations regardless of the architecture of a model.

1.2.2 Model-agnostic Explanations

To resolve this shortcoming, researchers have begun to focus on explainers that can theoretically explain any model. This idea means one can run the explainer on their classifier, whether it is a DNN model based on Long Short Term Memory (LSTM) [31] or a Support Vector Machine (SVM) that uses a specific kernel. In other words, one can train different classifiers on the same dataset while using the same explainer to get insights into each classifier's decision. A secondary advantage of such explainers is that they would allow users to adapt themselves to a specific kind of explanation, as they will not be using a separate explainer for each different classifier. We will discuss through this thesis that the way an explanation is framed to the user is a critical part of understanding the decision-making process of a black-box system.

Local Interpretable Model-agnostic Explanations (LIME) [59] is a recently published method that claims it can explain any model, regardless of its underlying architecture. At the time of writing this manuscript, it has received more than 3,700 citations since it was introduced in 2016 and is used in different domains. The authors of the paper use a few well-known datasets to attempt to prove their claim in their paper. They show how their method works on some instances of those datasets. However, a question remains: does it work on any dataset?

1.3 Thesis Statement

In this manuscript, we first show that while LIME claims to be a fully model-agnostic method, the reality is that it is not capable of explaining *any* black-box model. Besides, we show it does not perform well on all datasets.

We later show we can come up with a real model-agnostic method that is based on rules. That is an explanation that provides beyond a list of features, regardless of the model it tries to explain.

Using our solution of explainability (BARBE), we can build an interactive system where users are allowed to interact with the system to see how the predictions, together with the explanations, can change when the input

changes.

1.4 Thesis Contribution

This manuscript contains two major contributions, along with a few other minor contributions to the field of XAI.

We introduce Black-box Association Rule-Based Explanations (BARBE) as the main contribution. This model-agnostic explainer takes advantage of association rules to provide explanations for any black-box model when applied to tabular datasets. This framework, as we show in our experiments, outperforms LIME, a very popular explainer, in terms of the quality of explanations. Besides, BARBE can be applied to an extended class of classifiers. Furthermore, not only does BARBE present salient features as one way to explain the decisions of a black-box model, but it also provides rules as an alternative, yet a more human-understandable form of explanation. Finally, it also utilises a specific type of rules, namely association rules, to take into account the associations among different features.

We propose Black-box Association Rule-Based Interactive Explanations (BARBiE), an extension to BARBE, as the other major contribution of this manuscript. In BARBiE, we empower the user to interact with the explainer, to enable them to gain more trust in the black-box model. Interactive explanations of BARBiE allow the user to query the original instance as well as other data points in its proximity, letting them explore and understand how the behaviour of the black-box model varies in the vicinity of the instance. Additionally, it provides them with similar data points yet with a different class label.

Furthermore, we demonstrate with a few experiments that the explanations produced by LIME are not trustworthy among all datasets. Moreover, we show how the choice of the discretization approach affects the results. Finally, we utilise various metrics such as *Precision*, *Recall*, and an order-based similarity measure to evaluate and compare different explanation frameworks.

1.5 Thesis Outline

In Chapter 2, we will first review some of the fundamental concepts in XAI, such as why XAI has become a hot topic in AI, how we define explainability and interpretability and distinguish them, and briefly discuss the explainability-accuracy trade-off that has recently attracted many researcher's attention. Once we provided some definitions for interpretable and explainable systems, we then review some of the fully-interpretable models in short. We do so since they are needed for our experiments in later chapters. Explainable systems are the next topic we discuss in this chapter where we review different explanation approaches popular in the literature. Finally, we conclude this chapter by discussing the evaluation challenge in XAI: how explanations and explainable models should be evaluated.

Chapter 3 explains in detail LIME, one of the prominent explanation frameworks popular in the literature. We also describe a few alternatives to LIME that are model-agnostic explanation methods as well. In the final section of this chapter, we provide some experiments we conducted to evaluate the quality of the explanations of LIME quantitatively.

In Chapter 4, we review some of the rule-base classifiers popular in the literature. Rule-based classifiers are important to us since we take advantage of one of them in BARBE and BARBiE, the main contributions of this thesis. Once we reviewed prominent rule-based classifiers, we focus on a specific class of them: association rule-based classifiers, and we look into a few of them. Then, we discuss SigDirect, the associative classifier we picked for our work and discuss it in detail. Later, we provide reasons why we chose to add this classifier to BARBE and BARBiE over the others. We conclude this chapter by the experiment we did to show how our implementation of SigDirect performs compared to the original results reported by its authors.

We introduce BARBE in Chapter 5. Initially, we mention the shortcomings of other methods such as LIME. Then, we introduce BARBE and elaborate on how it differs from other methods and discuss its specifications. Afterwards, we discuss the details and settings under which we ran experiments to evaluate

BARBE. The next sections in this chapter contain various experiments we conducted to first fine-tune BARBE and then compare it against competing methods.

We propose BARBiE in Chapter 6 as an extension to BARBE. BARBiE provides the user with interactive explanations. These interactive explanations, in the form of "what-if analysis", allow the user to further trust the black-box model. We later show how BARBiE can help users get a better understanding of the system with "counter-factual" examples which BARBiE provides. Additionally, we discuss how it helps users create "editable" classifiers.

We finish this work in Chapter 7 where we conclude this research, and recommend some future work opportunities to explore.

Chapter 2

Background

In this chapter, we focus on Explainable Artificial Intelligence (XAI). We first look into why XAI is becoming so momentous, and then go over some definitions in the field. This is an important issue since, unfortunately, there are different viewpoints regarding concepts such as explanation. Later, we discuss some important topics within XAI, such as the so-called accuracy and interpretability trade-off. In the next section, we review some popular explainers that have been proposed by different researchers. Finally, in the last section we mention some of the methods researchers have used to evaluate different explainers.

2.1 Explainable Artificial Intelligence (XAI)

2.1.1 Why XAI Matters

In many applications of AI, justification is a must, and systems that do not have that capability are not considered for use at all. The medical domain, for instance, is a field that has greatly benefited from AI, yet it has so far not taken advantage of recent advancements such as deep neural networks, and this is mainly due to the lack of explainability in such systems [10]. Moreover, using Intelligent Systems (IS) in different domains has led to many unwanted results, such as classifiers that are unintentionally biased toward a specific class (because of an unbalanced train dataset). What follows are a few examples that show how significant and consequential the lack of explanations can be.

One example is when Amazon started using a system for recruiting job

applicants by processing their résumés using AI. After a while, they noticed that the recommendations provided by the system were biased toward male applicants. This critical problem in the system eventually led to disbanding the team who had developed the product [15]. If the team had used an explanation tool to notice the issue in the development phase rather than knowing it when the product was being used in production, they could have avoided the issue and built a better product. This is in addition to the damage caused to the company for being advertised as biased toward a specific gender.

In another case, Bloomberg.com reported that a system developed to recommend regions for Amazon's same-day Prime Delivery expansion was biased toward neighbourhoods dominated by white residents. Eventually, after receiving criticisms from different Congress members, Amazon announced they were adding other neighbourhoods in the cities to the program [2]. If the leadership team at Amazon had requested the development team to explain why their algorithm was suggesting these neighbourhoods, they could have avoided this issue.

One interesting research project revealed that a system that had been trained to detect colon cancer from patients' medical records was using non-medical features. While the system worked excellently, researchers soon noticed that the classification was based on the fact that all such patients had been dispatched to a specific clinic, and the name of that clinic was what was helping the AI system to detect them correctly not their medical information [67].

Finally, The European Union (EU) introduced a law called the General Data Protection Regulation (GDPR), which governs privacy of users' data. This law was put into effect in 2018 [24]. While this law mainly deals with data privacy and tries to force all companies dealing with EU residents' data to protect it, the law also requires them to provide explanations to their clients when an automated system such as an IS[1] make a decision for them that has legal effects. This is what is referred to as the *right to explanation*, which

[1]Please note that not all automated systems are intelligent systems yet this law applies to all such systems.

requires justification when an automated system decides for customers. In fact, a well-established requirement under the U.S. Equal Credit Opportunity Act [21] requires that credit companies provide relevant and insightful reasons when a person is denied credit. GDPR, however, extends to more categories.

The Defense Advanced Research Projects Agency (DARPA) announced a program called Explainable Artificial Intelligence (DARPA XAI) in 2016 [17] where the goal was to produce systems explainable to humans.

All these cases, from issues that private corporations such as Amazon have had, to the GDPR law enacted in the EU, have caused a significant surge in the need to provide explanations for the decisions made by IS. As we discussed in previous chapter, explaining decisions of AI systems to users to let them understand the rationale behind the decision is not a new concept in AI as it was an essential requirement of Expert Systems; however, the fact that AI currently dominates many different aspects of our lives (e.g., autonomous vehicles, virtual assistants, recommender systems) requires researchers and IS developers to further focus on providing explainability to users.

2.1.2 Definitions in XAI

There is no consensus on the definition of explainability and an explainable system among researchers, and even more interestingly, there are different views regarding interpretability and interpretable models. First, we review some definitions suggested by different researchers and then we conclude this section by providing our own definition.

Explainability and interpretability are alas sometimes used interchangeably in the literature [19], implying that any model providing some type of explanation is also interpretable.

Biran and Cotton [7] define the interpretability of a model to be the degree to which the human observer can understand the underlying causes of its decisions. While Miller [45] adheres to the definition of interpretability provided by Biran and Cotton, he states that he considers interpretability and explainability equal while Biran and Cotton consider explainability as one way to provide interpretability in addition to a system's introspection.

Lipton [40] defines interpretable models within two groups: transparent models, which to some degree are comprehensible to the user, and post-hoc interpretable models, which are systems that try to explain a black-box model using alternative approaches. The first group (i.e., transparent models) is categorized into three different levels:

1. Simultability: A user, given the data and parameters of the system, can calculate the final prediction of the system in a reasonable amount of time.

2. Decomposability: Each part or component of the system has a meaning to users, like nodes in a decision tree

3. Algorithmic: Users only understand the algorithmic features of the system such as shape of the error surface.

For example, a sparse linear model such as logistic regression fits into the first category because a user, provided with the internals of the system such as its coefficients, can compute the final output in a reasonable amount of time and also understand the full behaviour of the system. A deep neural network model, however, does not fit even into the third category since, for example, the shape of the error curve is not known. Finally, the author argues for the interpretability of textual and visual explanations and considers them among post-hoc interpretability techniques. Lipton characterizes them as interpretable since he believes humans could provide a similar type of interpretability (i.e., in a post-hoc form).

Doshi-velez and Kim [18] use a human-centred definition for interpretability, which is "the ability to explain or to present in understandable terms to humans".

As we can see, in all the definitions provided above, explainability is equivalent to interpretability; however, there are some authors that differentiate these concepts.

Montavon et al. [49] define interpretation as a mapping of an abstract concept to a domain that humans can understand such as images or text

while they consider an explanation as a set of features of the interpretable domain that have contributed to producing a decision.

Now that we have reviewed some definitions by different researchers, let us discuss it from a broader point of view where we take advantage of the definition of interpretability in logic [4].

Interpretation, explainability, and semantics are well-defined in mathematical logic (e.g., Mendelson [44]). In logic, the semantic interpretation of an expression is obtained by interpreting each component in it. In this way, assuming the vocabulary of the representation is precise, the user can interpret the expression compositionally by interpreting individual components of it. In our view, we can take advantage of the aforementioned definition for interpretation in logic and adopt it to machine learning (ML): an interpretable model is one that a human user can read or inspect, and analyze it in terms of its composable parts.

In this way, interpretability refers to a static property of the model, and can vary from fully interpretable (models such as a small decision tree), to deep neural network models in which interpretability is more complex and typically limited. For instance, consider what each layer learns in a convolutional neural network (CNN): early layers are responsible for extracting low-level features such as edges and simple shapes, while later layers usually extract high-level features whose semantics are understood with respect to a domain. In fact, with this perspective, models such as DNNs could hardly be classified as interpretable. It is important to point out that interpretability applies to the interpretation of a learned model before considering the inference the model can do. Note that we are against classifying models as interpretable or non-interpretable, but rather we believe there should be a spectrum allowing an interpretability score to be assigned to each model.

On the other hand, explainability has to deal with what kind of output the system provides to the user, rather than how a human user directly interprets the meaning of each model component. In other words, explanation has to deal with clarifying the reason or reasons a prediction was made or an action was taken. Thus, we define an explainable model as a system which is capable of

providing explanations without requiring any extra computation. *Explainability* is, thus a dynamic property of a model, in the sense that it requires run-time information to produce explanations. Explainability is about the mechanism of justification provided for an inference or prediction using a learned model, regardless of whether the model is clearly interpretable or loosely interpretable. Figure 2.1 illustrates the distinction between interpretability, which is about understanding a predictive model learned from data during training, and explainability, which relates to the clarification and justification of a prediction or decision made in the presence of a new observation.

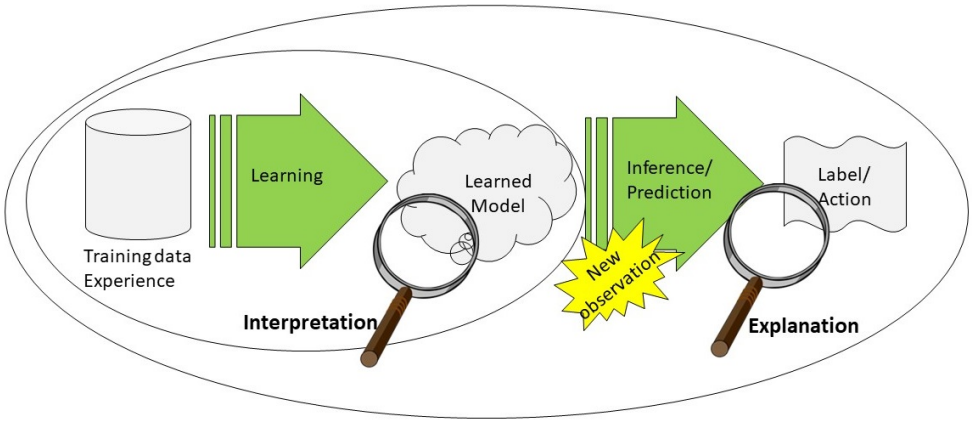

Figure 2.1: Interpretability of a model vs. Explainability of a prediction. From [4]

Based on the above definitions, models such as decision trees and rule-based systems that are considered fully-interpretable are also explainable, while deep models are not. For example, once we add an explanation module to the deep neural model (e.g., Babiker and Goebel [5]), they become explainable systems as well.

2.1.3 Accuracy-Explainability Trade-off

There is a popular belief that the explainability of models comes with a significant drop in their prediction accuracy. For example, authors of the DARPA XAI Broad Agency Announcement (BAA) [17] have implied their support for this in one of their figures. The figure, shown in Figure 2.2, depicts an inverse correlation between accuracy and explainability among different classifiers so

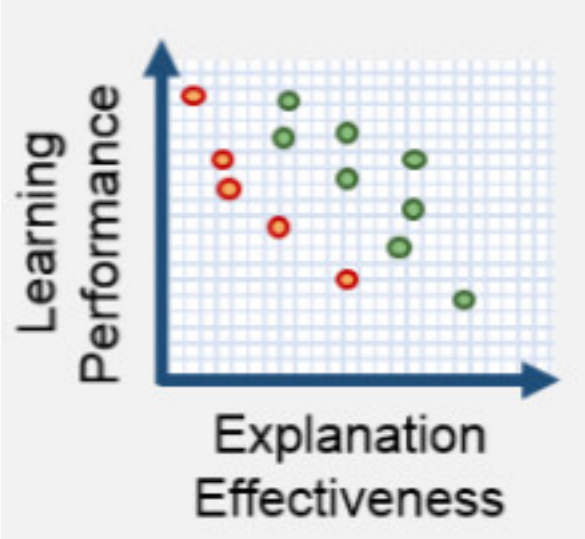

Figure 2.2: An imaginary figure by authors of DARPA BAA [17] that shows their support for the accuracy-explainability trade-off. In this figure, each red dot corresponds to one classifier while the green dots represent the same models yet with added explainability modules. The authors of the figure want to imply that the goal of XAI is to improve the explainability of models with a very limited impact on their learning performance.

that on a given dataset, one can either train a classifier with high accuracy but less meaningful explanations, or train a classifier that has effective and useful explanations yet is not able to achieve the learning performance of the other classifiers. Some research papers such as Ribeiro et al. [59], Lundberg et al. [42], and Lakkaraju et al. [36] also support this belief. This concept is what has led many to invest in explainers that try to explain black-box models so they can get both high accuracy and good explanations.

There are, however, researchers who disagree with the point mentioned above. Rudin [61] discusses in detail why, in her view, that belief is not solid. Her work describes the figure as an imaginary figure in which the axes have no units, and more importantly, such an illustrative difference between prediction accuracies of different classifiers is not the norm in data science applications.

2.2 Interpretable Systems

We review different types of interpretable classification systems. We mainly focus on reviewing some models considered transparent by most researchers

(i.e., humans can fully understand how they work, and are toward the fully-interpretable end of the interpretability spectrum). Afterwards, we take a look at the required criteria for transparent models. The reason we are interested in transparent models is that we are going to use them in our experiments in Chapter 5.

2.2.1 Transparent Models

In this section, we go over transparent models and briefly describe them. Many different models have been developed over the past few decades. These models include, but are not limited to, linear regression, logistic regression, decision trees, random forest, Support Vector Machine, and Deep Neural Networks. There is, however, consensus on the transparency of only three of them [26]: Generalised Linear Models, Decision Trees, and Rule-based models. Moreover, even for these classifiers, some conditions must be met in order to make them or consider them transparent.

Generalised Linear Models

Generalised linear models [53] are among the class of interpretable models. They include classifiers such as logistic regression and regression models, like linear regression. Linear regression builds a model in which its goal is to find a linear relationship between the value of a dependent variable with one or more independent variables that are provided. In logistic regression compared to linear regression, the dependent variable becomes binary, and a logistic function is used to map the log-odds to a probability value (as desired for the dependent variable).

In these two forms, we can use the weights of features as the interpretation of the system either directly or indirectly[2]. For linear regression, the weight directly indicates the importance of that feature in the final score or value generated by the system. With logistic regression models, on the other hand, we can use the weights to calculate the odds ratio for each feature [48].

[2]Please note that we assume we are working with a standardised data; otherwise, we also need to include the standard deviation of each feature when computing their respective importance.

Equation 2.1 represents the *log odds* of a logistic regression model trained to classify banknotes as either genuine or forged[3]. $x1$, $x2$, $x3$, and $x4$ correspond to the variance, the skewness, and the kurtosis of the wavelet transformed image, and the entropy of the image, respectively. Finally, Equation 2.2 shows the final equation needed to compute the probability of the banknote being genuine or forged. As you can see, We can easily understand how the system is computing the final class label, and the importance of each feature is evident.

$$log\ odds = -9.019 \times x1 + 0.705 \times x2 + -0.871 \times x3 + 6.1769 \times x4 \quad (2.1)$$

$$p(genuine) = \frac{b^{log\ odds}}{b^{log\ odds} + 1} \quad (2.2)$$

Decision Trees

Decision trees are another type of transparent models. A decision tree model trained to classify breast tumours as malign or benign is depicted in Figure 2.3[4]. While mathematical equations (e.g., Equation 2.2) could be used to describe generalised linear models, decision trees are better understood by visualizing them as a tree. Unlike linear models, decision trees divide the space into multiple separate parts. Each node in the tree is used to further split the current sub-space into smaller segments, making smaller clusters of population. Each path in the tree corresponds to a different partition in the domain space, as the value of at least one feature is different compared to any other partition. Different algorithms have been proposed over the years to achieve such a tree. One very famous and popular decision tree algorithm is C4.5 [57], which is widely used in different domains. In order to create the tree at training time, C4.5 uses the difference of entropy as the criterion to decide which feature should be used when expanding a node. At test time, we only need to see the label of the leaf node of the path applying to the instance,

[3]UCI banknote authentication dataset is used for this experiment.
[4]UCI breast cancer dataset is used in this figure.

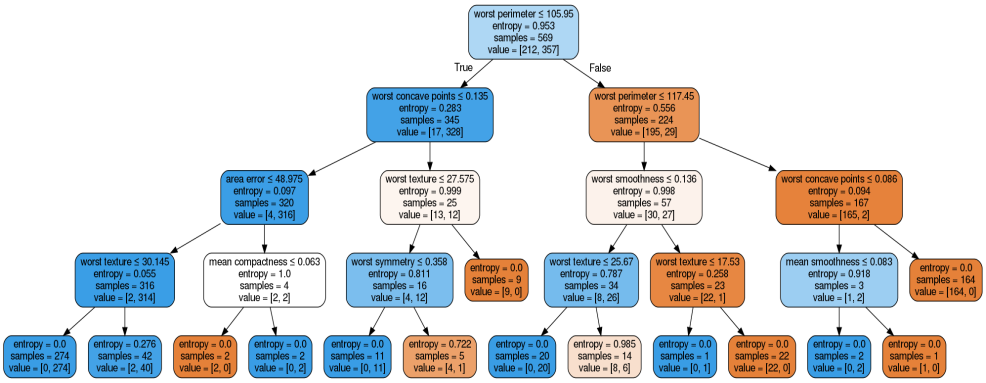

Figure 2.3: A decision tree for classifying breast tumours as malign or benign.

originating from the root node. The leaf node corresponds to the class of the instance. Therefore, the path and the features used along the path can be presented to the user, which helps them understand why and how the model has come to the conclusion. The path is the explanation. Furthermore, the order of the features present in internal nodes can give more insight about the system. If a feature occurs in upper layers of a tree, we can infer that it has a high global importance, while a feature used in a node in lower layers implies it is locally important to that subspace.

Rule-based Models

In addition to linear models and decision trees, researchers also agree on the transparency of rule-based models. The main idea in this type of classifiers is to come up with a set of patterns that are frequent in the train set and then exploit them at prediction time to classify the unseen instances. Each rule contains a set of antecedent items (each corresponding to a feature), and a consequent, which is a class label. Given these rules, the final class is determined using the criteria that is unique to the method; for example, one approach for confidence-based methods would be to use the sum of the confidence score of rules of each class to determine the final class in a classification task. In order to interpret such systems, a user can look at the rules that apply to the individual instance. We take a closer look at this type of models in Chapter 4. The rule-based models, unlike decision trees, can provide users with more than one explanation because each rule is an explanation.

Interpretability Criteria

Although most researchers agree on the interpretability of the previously mentioned types of models, there are some conditions that many believe these predictors need to have so that they can be understood by humans. Generalised linear models should be sparse. If our dataset contains hundreds of features and the final model has a nonzero weight for each feature, the user is not able to interpret it correctly. In the case of tree-based models, a similar problem exists: if the tree has many layers for most branches (meaning that there are many features involved for inference), it would be difficult for a user to understand. Finally, for rule-based methods, if the number of generated rules is very large, a user cannot fully comprehend which rule had more impact on the outcome, and thus is unable to interpret the system thoroughly.

This issue has led to the development and use of some techniques or methods that try to make these models more interpretable. For example, a stronger regularisation could be enforced to zero out the effect of some features when training a logistic regression model, resulting in fewer features involved in the classification task (We have used this technique in Equation 2.2 to limit present features). In decision trees, one could easily limit the number of layers of the tree to get fewer features involved in the decision-making process (as could be seen in Figure 2.3 where we limited the tree depth to four). In rule-based methods, there are pruning techniques that try to reduce the number of generated rules while maintaining the overall accuracy of the model. In fact, to some degree such methods help avoid over-fitting the training data.

Having said that, we should also emphasise that we can still consider the original approaches (i.e., without the modifications mentioned above) interpretable to some degree, since people's capabilities to understand and interpret models are different, and it is a subjective matter. We cannot merely specify an explicit criterion to make a method interpretable or opaque.

2.2.2 Other Interpretable Models

As we discussed in Section 2.1.2, we believe there is a spectrum of interpretable models. In this spectrum, in contrast with models such as small decision trees or rule-based models, DNNs should be placed in the other end of the spectrum. Models such as Random Forests (RF), however, may be put somewhere in the middle of the spectrum depending on a few factors such as the number of trees in the model, and the depth of each one. These can significantly affect where each individual RF model should be placed. If there are hundreds of trees in an RF model, for example, it should be placed in the less-interpretable end, close to DNNs.

2.3 Explainable Systems

The first class of explainable systems is called model-specific explainers. All classifiers in this class have a built-in explanation segment that provides some explanation to the user on the whole model's reasoning, thus making the whole model explainable. In the second class, however, an external explainer is added to the black-box model so that it can provide an explanation for the user. This group of models, where the research focus is on the external explainers and not the black box models—since the models lack any explainability feature themselves—are called model-agnostic explainers. These two groups are also called model-dependent and model-independent, respectively, as well. We should emphasise here that this dependency or specificity is in regard to having access to the internals of the model. There is another way to classify explanation techniques where models are grouped based on their knowledge of the domain [4]. Based on this classification, an application-dependent explainer assumes the user has the required knowledge about the domain thus it employs the domain vocabulary for explanation (e.g., using medical terms in medical applications). A generic explainer, on the other hand, does not make that assumption and thus provides explanations that lack any knowledge from the domain.

Finally, it is noteworthy that while training and using a transparent global

surrogate model can be an alternative approach in order to provide explainability, we do not cover them here since they basically are equivalent to transparent models in terms of explainability. In fact, they usually are one type of a transparent model, such as a decision tree or a rule-based method learned from the black-box. For example, Craven and Shavlik developed TREPAN in 1996, which created a decision tree that mimicked the behaviour of a neural network [14]; Núñez et al. [54] proposed a method to create rules from a Support Vector Machine. All of the post-hoc explainers, unlike transparent surrogate models, provide explanations for each instance (data point) and are sometimes called *Outcome Explanation* in the literature [26].

2.3.1 Model-specific Explainers

In this section, we briefly discuss model-specific explainers and review a few of them. These methods are also sometimes called model-dependent in the literature. The main feature of these explanation methods is that they are specifically created to work on a particular architecture. While most of these methods are designed to explain DNNs, there are some methods that are developed to explain other classifiers. Below, we review a method that works on tree ensembles and then we go over few of methods developed to explain DNNs.

Explaining Tree Ensembles

Tree ensembles are a group of classifiers that work by training and using multiple decision trees. This combination can achieve a higher accuracy score compared to a single decision tree and is hence generally preferred. One side-effect of this improvement in accuracy is, however, the lack of interpretability in ensemble models. If in a task, the accuracy is the critical criterion and understanding the way the model came to that conclusion does not play any significant role, it is reasonable to opt-in for ensemble methods. However, if understanding that rationale is critical, ensemble methods would not be the best choice unless we add explainability to them.

In their paper, Moore et al. [50] provide a method to produce explanations

for each instance in tree ensemble methods such as random forests. Initially, they compute the change in the expected output for each node in all the trees of the ensemble model. Afterwards, they leverage them to produce an explanation for any given data point. The explanation provided by their system contains an importance score for each feature present in the data point. This score is the sum of the changes in the prediction output of all corresponding nodes along the paths used to predict the instance's class label.

Attention-based Approaches for Explaining DNNs

In recent years, most researchers have focused on explaining DNNs. This is due to the huge popularity of these methods, that they have been applied to various domains from sentiment analysis in Natural Language Processing (NLP) to object localization in images.

Attention mechanism [6] is one of the recent breakthroughs that has changed the performance of DNNs, especially in the field of NLP. Before the introduction of attention mechanism, DNNs relied on Long Short-Term Memory (LSTM) architecture to avoid the vanishing or exploding gradients problem in sequence to sequence tasks such as Neural Machine Translation (NMT). In NMT, a sentence in a source language is translated into a sentence in a target language using a deep neural architecture. The decoder in an LSTM-based NMT model has to use the information in the state vector of the last word in the source sentence to generate the first word in the target sentence. As one can expect, however, information corresponding to the first word in the target language usually lies at the beginning of the source sentence. Having to find the information in the whole sentence, rather than a few words, makes the translation task more challenging. The attention mechanism was introduced to overcome this weakness. When taking advantage of the attention mechanism, not only does each word in the target language benefit from the state vector of the previous word (as it is a recurrent architecture), but it also takes advantage of the weighted sum of the state vectors of source words. The aforementioned weights essentially specify the source words the model should pay more attention. The introduction of this mechanism allowed researchers

to obtain better performance in different tasks such as NMT [6].

One of the benefits of using the attention mechanism is that it can act as another way to give a hint about the internals of the black-box system to the user.

The method implemented in [12] is an example of using the attention mechanism in the medical domain. In their work, Choi et al. use a DNN model to process Electronic Health Records (EHR) in reverse time order, and predict a future diagnosis of heart failure. Their model processes EHR as a time series dataset and uses two attention mechanisms to process them. Additionally, they rely on the weights in the two attention layers in the model to provide explanations such as which visits to a physician played a more significant rule in their decision. The record that the multiplication of the attention weights by its embedding weights results in the maximum value, for example, is the most important one. Although many research articles benefit from attention mechanism to provide explanations, some researchers disagree and argue against using attention weights as a way to explain decisions [35].

Gradient-based approaches for Explaining DNNs

In addition to attention mechanism, computing the gradients is another way to exploit the internals of deep learned models. In this approach, the idea is to compute a gradient with respect to the predicted class and use the back-propagation algorithm to propagate the gradient to the input. Afterwards, one can combine the input with the gradient to capture the salient pixels which can be used to explain the predicted class (e.g., Grad-CAM [64]).

2.3.2 Model-agnostic Explainers

This part contains a brief overview of methods that claim they can explain decisions made by different architectures. As we mentioned in the last chapter, this approach has the advantage of being able to explain different architectures. However, we also should emphasise that getting correct explanations is a big challenge, due to the fact that the explainer has no access to the internals of the system, while a model-specific explainer has access to every component of

it.

Local Interpretable Model-agnostic Explanations (LIME)

LIME [59] is the most famous model-agnostic approach used to explain black-box models. It works by creating perturbed samples and then training an interpretable model using such samples. We discuss LIME, and a few alternative approaches similar to LIME in detail in the next chapter (Chapter 3).

SHAP

SHAP framework [42] is based on Shapley value [65]. Shapley value originates from cooperative Game Theory. In cooperative games, the group of players tries to maximize the prize in the game, and in the end, the reward is distributed among the players. Shapley value tries to answer the question of how to distribute the reward between players based on their contributions, and it has proven to be the answer to this problem of reward assignment. The same concept could be used to explain the decisions of classifying individual instances of a dataset. Different features of an instance correspond to the players of the game, while the feature importance corresponds to the reward distributed at the end of the game to a player. The formula to compute the reward for each feature is as follows:

$$\phi_i(val) = \sum_{S \subseteq \{x_1,\ldots,x_p\} \setminus \{x_i\}} \frac{|S|!(p-|S|-1)!}{p!}(val(S \cup x_i) - val(S)) \qquad (2.3)$$

In this formula, given an instance x containing p features, to obtain the feature importance of feature i, we shall compute a weighted sum over all subsets of features of x that exclude i. In this sum, we want to measure how the addition of i to a subset affects the final prediction of the model. The weight of each subset equates to the number of various ways that the subset could have been generated.

While this method looks ideal to use as a way to explain instances, it is very computationally costly as it needs to consider all subsets in the sum. As a result, authors of SHAP suggest sampling to make it computationally efficient.

Anchor

Another approach introduced after LIME is called Anchor [60]. In their work, Ribeiro et al. (the same authors as of LIME) point out that a weakness of models like LIME is that the explanation comes from a linear model. They show that an importance score for each feature cannot work all the time. As an example, the authors provide a text classification example (Figure 2.4) where the word "not" can have both positive and negative influence depending on the other words in the sentence. This dependency cannot be shown in an approach like LIME, as it assumes feature independence and produces individual feature importance scores. To overcome this shortcoming, they suggest providing a set of salient features in the form of "if-then" rule as the explanation. They call such rules *Anchors* since they anchor the prediction toward the target class, and consequently, modifying the rest of the features would not alter the class label. Each anchor, in their framework, is essentially a set of important features such as "not" and "bad". They argue for the high precision of these anchors in their work. High precision means when taking advantage of an anchor, most instances where this anchor applies to should be classified correctly. For example, if they have an anchor such as "not" and "bad" $\longrightarrow$ positive sentiment, precision measures how many instances of a given dataset are classified correctly using this rule).

They employ a greedy algorithm to construct an anchor. In their algorithm, they add one feature to the anchor in each iteration, where the generated anchor provides the highest precision among candidates. They rely on a synthetic set of instances in that vicinity to compute the precision. Unlike LIME, however, they try to achieve the least number of calls to the black-box model by finding the smallest required instances using a multi-armed bandit algorithm. By taking advantage of this approach, they adaptively generate the neighbourhood around the instance. Finally, they terminate the greedy algorithm once the precision is above a heuristically set threshold (they set it to 0.95 in their paper).

A shortcoming with this approach is that it does not reveal the associations

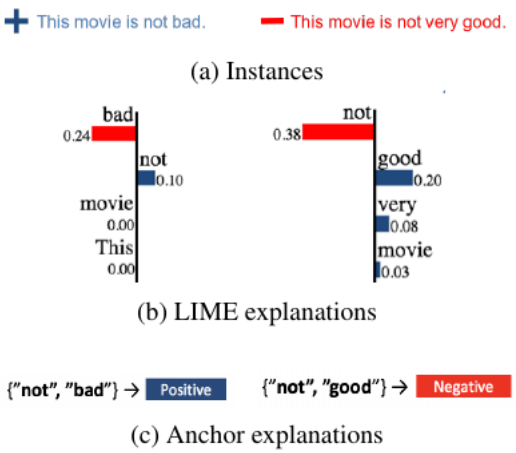

Figure 2.4: A figure taken from [60] where the authors show different explanations produced by LIME and Anchor. a: 2 instances of a sentiment analysis task, b: what LIME outputs, c: Anchor output where the combination of words have the influence.

among features. Additionally, in contrast to LIME, it cannot provide any relative feature importance scores anymore.

2.4 Evaluating Explainers

In order to use or propose a new explainer, authors need to evaluate their method and this is the norm for any proposition in ML. In XAI, however, things work slightly differently since there is even no consensus on the definition of explanations.

This is in fact in contrast with showing the superiority of a new classifier or a new architecture in a deep learned model which only requires showing its performance in terms of certain metrics, such as accuracy or F-score. It is, however, challenging to benefit from the same metrics for an explainer as there is usually no ground truth to compare against.

2.4.1 Proposed approaches

Miller [45] discusses explainability from a social science viewpoint and emphasises on simplicity of the explanation, together with its generality and coherence with prior belief. In his view, providing explanations that have

fewer causes yet could cover more events are better than providing detailed explanations with multiple causes altogether. He backs his view by showing an example where respondents to a survey preferred simpler causes over more complex ones. He also positions himself as an opponent of using statistical relationships to explain events and argues in favour of the interest in causes by end-users rather than associative relationships. He further claims that the most likely cause is not always the best explanation since they may be presupposed by end-users. He, however, does not provide any quantitative way to analyze explanations and his view mainly takes causal explanations into consideration and leaves correlation-based ML systems aside.

Another work that suggests a taxonomy for evaluating explanations is the work done by Doshi-Velez and Kim [18]. In their work, they propose three different groups of evaluation approaches similar to the ones used in ML. They propose application-grounded evaluation in the first approach where explanations are evaluated in real tasks using experts. The second approach, called human-grounded metrics, also benefits from humans in the experiments while limiting the tasks to simplified ones (compared the actual tasks in the previous approach). Functionality-grounded evaluation is the third approach which does not involve humans but uses automated experiments on proxy tasks.

2.4.2 Evaluation Approaches in practice

In addition to using automated experiments, many papers rely on experiments that involve humans in the experiment. Take Grad-CAM [64] for example. In one experiment, they measure how removing particular image patches affects the prediction scores of the CNN model (i.e., image occlusion), and notice the biggest change occurs when they remove patches that were part of the explanation produced by their method. Another experiment involves creating class labels and bounding boxes out of their heat-maps for images from the ImageNet Localization challenge [16], and then comparing them with the ground truth labels and bounding boxes obtained from the challenge. They rely on humans in other experiments. In one experiment, they provide users with visualizations produced by different explanation techniques and ask them to rank

the visualizations based on their informativeness to find out about their trustworthiness. As you can see, the authors take advantage of various experiments to, however, indirectly evaluate their method.

2.4.3 A Unique approach for model-independent explainers

Another way to quantitatively evaluate explanations of a model-independent explanation framework is to replace the black-box model with a fully-transparent one. As we mentioned in Section 2.1.2, fully-transparent models are also explainable systems. In this method, we can look into the model and extract the ground truth explanations and compare with the explanations. Take a small decision tree as an example. Given an instance, the decision tree's inference algorithm determines the label based on the leaf associated with the instance. If we take the path leading to that leaf, we can extract the features present in that path and return them as the explanation. Such a set of features can be considered the correct explanation as they are unquestionably the features the model has used to label the instance. Using this approach, we can evaluate the model-independent explainer by finding how many features it categorized as important were present in the fully-transparent model (i.e., precision score), or how many of those features were present in the explanation provided by the explainer (i.e., recall score).

To check the trustworthiness of its explanations, authors of LIME use a linear model and a decision tree to evaluate their framework on two different datasets. The results they report on decision trees outperform the ones from logistic regression models That is interesting since LIME framework benefits from a linear model (linear regression) as its interpretable model and one can expect better results when the black-box model is also a linear model. In our work, to further investigate the performance of LIME, we use a decision tree as we present in Chapter 3. We take advantage of the same classifier when evaluating our own approach 5 since it is also a different classifier than what we use in our method (an associative classifier). Besides, it allows us to have a fair comparison against LIME.

2.4.4 Wrap up

As the discussion shows, despite providing many different approaches to tackle the evaluation problem in XAI, very limited approaches have been adopted in practice. This is mainly due to the huge gap between what those conceptual papers define as an explanation, and what current methods produce as explanations. The majority of current explanation systems provide explanations in the form of heat-maps [5], [64], feature importance scores [42], [59], or some rules [25], [60]. In all these cases, what matters the most is the trustworthiness of the explanations (i.e., is it really the rationale behind the model's decision?). Matters such as coherence with prior belief are yet to be incorporated into evaluations. This mainly has to do with the fact that current explanations are very limited and evaluating explanations based on their coherence with prior belief, for example, requires richer explanations. Refer to the work of Atakishiyev et al. [4] for a full discussion on different levels of explanations.

Chapter 3

Local Interpretable Model-Agnostic Explanations

In this chapter, we review the Local Interpretable Model-agnostic Explanations (LIME) method in more detail. We first discuss how it works and then illustrate its explanations on some different tasks. We conclude this chapter by reviewing a few alternative methods based on LIME.

3.1 What is LIME?

LIME is a framework that was published in the Knowledge Discovery and Data Mining (KDD) 2016 conference by Marco T. Ribeiro, Sameer Singh, and Carlos Guestrin. Their method has received a massive welcome from the research community. Since it was one of the early frameworks claiming to be model-agnostic, it was adopted by researchers in different fields. Moreover, due to its availability as a package in Python programming language, many people have started using it off-the-shelf to add explainability to their systems. For example, Whitmore et al. [71] applied LIME on 2-D chemical structures. They used it on a black-box model to understand the classification of biologically produced fuel compounds. LIME can also be applied to analysing music content [46]. Credit assessment is also a domain in which explanation is very critical, and as a result, numerous publications in this topic benefit from LIME in their work [47], [52]. Not only has LIME been applied to different domains, but it also has been used in various tasks. Besides classification,

which composes most of the literature and what the original LIME supports, it has also been applied on the Named-Entity Recognition (NER) task, a sequence to sequence tagging task [68]. NER is a task in Natural Language Processing (NLP) that when given a sequence of tokens (the sequence usually is a sentence), the goal is to find and annotate named entities, such as names of people (e.g., Noam Chomsky) or organizations (e.g., University of Alberta) in the text with proper tags. Authors of [68] propose two different approaches for using LIME. In their first method, they provide explanations for each word in the sentence. In their second approach, however, they try to explain each named-entity (as a phrase rather than a word) using LIME.

As we mentioned in the previous chapter, LIME should be considered as a post-hoc explainer that can be used to explain individual decisions of a classifier. As its name suggests, it claims to be model-agnostic and capable of handling any black-box model.

3.2 Generated Explanations

In this part, we take a look at the output LIME generates and the way it conveys the explanation it has generated to the user.

LIME, similar to other post-hoc model-agnostic approaches, has to rely on the input and output of a model in order to generate explanations, since it assumes the classifier acts as a black-box and thus, cannot provide any further information. In other words, unlike transparent models where at least a portion of the decision tree, if not the whole tree, could be visualised, or some rules in a rule-based method could be presented, LIME has no information regarding the internal structure of the model: the input space and the target predictions are its only available sources of information. We should again emphasise the fact that explanations are for each data point and not for the whole model.

The input for general machine learning problems (e.g., classifying mushrooms based on the shape of the cap, and their odor as edible or poisonous) is usually in the form of some categorical or numerical features. For such tasks,

```
cap-shape                      convex
cap-surface                     scaly
cap-color                      purple
bruises?                           no
odor                             none
gill-attachment                  free
gill-spacing                    close
gill-size                      narrow
gill-color                  chocolate
stalk-shape                 enlarging
stalk-root                    bulbous
stalk-surface-above-ring       smooth
stalk-surface-below-ring      fibrous
stalk-color-above-ring          white
stalk-color-below-ring          white
veil-type                     partial
veil-color                      white
ring-number                       one
ring-type                     flaring
spore-print-color           chocolate
population                   solitary
habitat                         woods
```

Figure 3.1: An instance of mushroom dataset

LIME's explanations are a score assigned to each feature (either positive or negative). For binary classification tasks where we only have two classes, positive numbers correspond to one class while negative numbers correspond to the other class. For multi-class classification, LIME utilises a one-vs-all approach and produces different explanations for each class. In Figure 3.1, we show an instance from the mushroom dataset [20], which lists attributes of an edible mushroom. In Figure 3.2, we show LIME's output regarding that instance. As shown in the figure, LIME explanation suggests that while having a narrow gill size or a close gill spacing makes a mushroom look like a poisonous one, having no odor and a smooth stalk above the ring outweigh those attributes and make the mushroom an edible one.

For tasks that apply classification on natural text, such as sentiment analysis, individual tokens are usually transferred to a vector space (e.g., n-gram vectors or word-embeddings). LIME explanation, however, uses the tokens themselves and outputs some of the words as the explanation rather than just numbers associated with them.

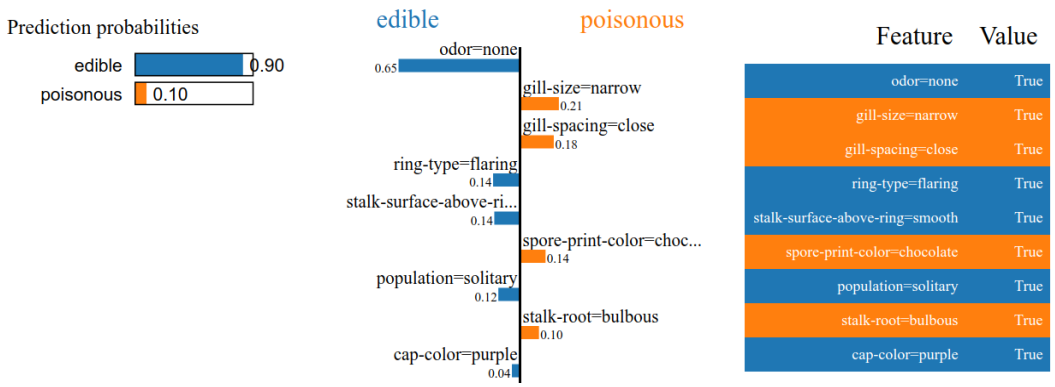

Figure 3.2: *Explanation* provided by LIME for the instance in Fig. 3.1 The probabilities on the left side are the prediction probabilities generated by the original model (the black-box system). In the centre, features and their corresponding importance scores sorted based on their importance generated by LIME, are shown. A greater absolute value in this table implies that LIME believes it has played a bigger role in classification. Finally, the right side figure shows the feature-values and their corresponding values, sorted based on their ranks. In the case of categorical feature-values, they are True or False, but for numerical features, it would show the value of that feature in the instance (e.g., 26 for age).

3.3 How does LIME work?

In short, LIME creates a synthetic dataset in the vicinity of the original data point, and after labelling its instances using the black-box model. It then uses it to create an interpretable model to produce an explanation for the data point. In this section, we look at the way LIME creates explanations in more detail. We first discuss how it creates a neighbourhood and then see how it builds the model. Finally, we review the way it generates the explanations from built model.

3.3.1 Creating sample data for neighbourhood

For this first step, LIME needs to create a new dataset so that it can use it later to train a transparent model. This new dataset contains new samples that are perturbations of the original instance, which the system should explain to the user. The perturbation technique is considered to be the most common approach used in model-agnostic explainers [42].

Although perturbation is the general idea behind the creation of the sam-

ples in this step, LIME utilises distinct approaches to generate samples among different types of datasets. For tabular data, LIME creates 5,000 new data points (the default value used in the Python package) unless the user sets the parameter to a different number. To construct these new data points, LIME creates that many values for each feature separately, and then randomly joins them. Perturbation for categorical features takes advantage of their frequency in the training data, where values that occur more frequently in the training data are also more likely to be present in the synthetic data. Regarding numerical features, the perturbation step relies on their proximity to the value of the feature in the original data point. As a result, it is more likely to have values that are similar to the original data point than different ones. It is also worth mentioning that LIME takes into account the proximity of the original data point and the newly created instances. This proximity is later used as a weight when training the local model. We discuss the technical details in Section 3.3.4.

3.3.2 Building Model for Neighbourhood

Once LIME has the generated neighbourhood, it trains a transparent model that can be used in the next step to provide explanations to the user. In this step, the goal is to build a learned model that mimics the behaviour of the global model in the vicinity of the instance. LIME achieves this goal using the sampled data points that were created in the previous step together with their corresponding distances from the original data point. It then runs the original black-box model on all the sampled instances and obtains their labels. Using those labels, and taking into account their proximity to the original instance (i.e., how many features have different values than the original data), LIME trains a model. Here the model can be any of the transparent models, such as decision trees or linear models; however, LIME's authors use linear models in their paper and the provided Python package.

3.3.3 Generating Explanations from Local Model

Once LIME builds the local interpretable model, the next step is to extract and provide feature-based explanations to the user. This step is, in fact, the most straightforward step in the LIME algorithm: it only needs to extract the weights of each feature from the linear models, but even presenting the model (as it is) would still count as the desired explanation. The original paper only provides an example with linear models.

3.3.4 Technical Details

Now, let us take a closer look at LIME with the mathematical details. There is a black-box model f and an instance x that the model has classified. The instance has d features and can be shown as $x \in \mathbb{R}^d$. As mentioned earlier, we are also interested in a representation understandable by humans (such as words rather than word embeddings); as a result, LIME defines x' to be a human-friendly representation of x; that is $x' \in \{0,1\}^{d'}$ where d' is the number of features in x' transformed from d specially for each type of dataset. For images, for example, this can be super-pixels obtained by running a super-pixel segmentation algorithm on the original image. Using this binarization of the new features, LIME can choose which features are present and which ones are not, and hence further simplify the perturbation process.

The goal is to create a model $g \in G$ (G being the class of interpretable models) that has high fidelity to f in the neighbourhood of x. The domain of g is $\{0,1\}^{d'}$. In other words, the human-understandable form of x is used to when creating g. The goal mentioned above is achieved by building a synthetic dataset around x' by perturbing it and then train an interpretable model g using the new dataset.

The authors of LIME introduce two measures required for a good explanation: 1) Local fidelity: a model faithful to the global model in the neighbourhood of x; and 2) Model Interpretability: a model that is simple and easy enough for humans to understand. While the first measure is a conventional task of minimization of loss function in machine learning, the latter differs.

For the second measure, the authors suggest the number of layers for decision trees or the number of nonzero features in linear models as the measure of interpretability: the deeper a decision tree, the less interpretable it is. Finally, in order to train a model with these two measures in mind, they propose the following formula:

$$\xi = \underset{g \in G}{\operatorname{argmin}} \mathscr{L}(f, g, \pi_x) + \Omega(g) \tag{3.1}$$

In equation 3.1, $\Omega(g)$ is the measure of interpretability discussed earlier. On the other hand, $\mathscr{L}(f, g, \pi_x)$ is a loss function that measures the faithfulness of the learned model in relation to the global model. In other words, it calculates how accurate g approximates f in the vicinity of x. $\mathscr{L}$ is based on π_x, which is the importance of each sampled data point with respect to x (which is based on their normalized proximities).

The authors also define the loss function they use for linear models ($g(z') = w_g.z'$) in the paper:

$$\mathscr{L}(f, g, \pi_x) = \sum_{z, z' \in Z} \pi_x(z)(f(z) - g(z'))^2 \tag{3.2}$$

where π_x is computed in this way:

$$\pi_x(z) = exp(-D(x, z)^2/\sigma^2) \tag{3.3}$$

In this formula, D is a distance function unique for different types of data such as tabular or text. As an example, they suggest using L^2 as the distance measure for tabular data while recommending cosine similarity for text datasets. Finally, they apply a kernel smoother on the distance so that closer sampled data points have a higher impact on the total loss.

There is, however, no further discussion of the appropriate width of the kernel smoother in the paper. When we dig into the code, we find that its default value is $\sigma = 0.75$, which shows that different widths lead to totally different results [48].

3.4 Submodular Pick LIME

Submodular Pick LIME or SP-LIME, for short, is a method suggested in the LIME paper that tries to allow the user to understand the model by providing the most explanatory data points. In other words, it is an alternative tool to evaluate the trustworthiness of the system instead of the traditional global surrogate models where an interpretable model such as a decision tree is trained to mimic the behaviour of the complex model globally.

The user should provide the system with their budget, B. This budget corresponds to the number of instances the user wants SP-LIME to explain. Then, given some data points, SP-LIME finds the most informative ones by first running LIME on each one of them. Once LIME has run on all of them, it provides a feature importance score for each feature in each data point as the explanation. Using these values, a unique global weight is assigned to each feature, showing how important that feature is compared to other features. Finally, using these scores, a greedy approach is applied to find the most informative data points out of the given ones. These points are what SP-LIME returns to explain the global model.

The most critical part in a procedure like the one described above is, of course, the way the algorithm ranks and adds all data points to the set of chosen data points. SP-LIME suggests the addition of new data points to the set in a greedy way. This greedy algorithm, shown in Algorithm 1[1], chooses the instance that maximizes the marginal *coverage* in each iteration. Coverage (Eq. 3.4) is defined as the total importance of features that appear in at least one of the instances of the set.

$$coverage(V, W, I) = \sum_{j=1}^{d'} \mathbb{1}_{[\exists i \in V\,:\, W_{ij} > 0]} I_j \tag{3.4}$$

[1]Based on Algorithm 2 of LIME paper, with some changes

Algorithm 1 Submodular Pick LIME (SP-LIME)

Require: Instances X, Budget B
Ensure: Set V containing best instances
  for all $x_i \in X$ **do**
    $W_i \leftarrow LIME(x_i)$ ▷ Run LIME on this instance and get the explanation
  end for
  for all $j \in \{1 \dots d'\}$ **do**
    $I_j \leftarrow \sqrt{\sum_{i=1}^{n} |W_{ij}|}$ ▷ Compute global feature importance
  end for
  $V \leftarrow \{\}$
  while $|V| \leq B$ **do**
    $V \leftarrow V \cup \underset{x_i \in X}{\operatorname{argmax}}\ coverage(V \cup x_i, W, I)$ ▷ Pick the best instance
  end while
  return V

3.5 Modifications to LIME

While many people have used LIME in their systems, many people, including the authors of LIME, have created newer models to improve certain aspects of LIME. In this section, we first discuss KLIME [27], and LIME-SUP [33] which are essentially using LIME under the hood. We then review LORE [25], a method that exploits Genetic Algorithm for building the neighbourhood around the instance.

3.5.1 KLIME

KLIME is another variation of LIME developed by Hall et al. [27] and is available as part of the H2O Driverless AI platform.

KLIME's goal is to build explanations for a limited number of representative points so the explanations could be used for the rest of the points. This method, unlike LIME, segments the training data into K clusters and then trains a Generalised Linear Model (GLM) for each of the K clusters solely based on that cluster's data points. It currently uses the K-means clustering algorithm to create clusters. Moreover, it trains another GLM based on all the training instances of the dataset. To explain a new instance, if its corresponding cluster contains 20 or more training data points, the GLM for that cluster

is used to provide the explanation. If that number is less than 20, the global GLM is used. In this platform, K is computed in a way that the number of training instances correctly predicted using their corresponding GLM model is maximized.

One big issue with this approach is that clustering algorithms such as K-means are not stable and different initialization settings result in different clusters.

3.5.2 LIME-SUP

Another method similar to KLIME is the method developed by Hu et al. [33]. In contrast to KLIME, LIME-SUP uses supervised partitioning to split the training data. They use the idea of model-based trees in which nodes, unlike decision trees, are models themselves that only apply to a portion of the problem space. As a result, by splitting the space according to the training data, the local models available in leaves are more accurate compared to KLIME (they have less Mean Squared Error (MSE)). Additionally, the subspaces are more stable than the ones created by K-means in KLIME. Nevertheless, this approach requires the training labels to create the tree which may not be available at all times.

Although LIME-SUP improves upon KLIME, both methods are alternatives to speed up the explanation generation process at the expense of providing explanations from nearby data points.

3.5.3 LORE

As we mentioned earlier, rules are considered one of the most interpretable approaches to explainability and thus can be a perfect way to provide explanations to a user. In their work, Guidotti et al. [25] take advantage of rules for providing explanations. In their method, they create a neighbourhood around the instance using a Genetic Algorithm. Moreover, they enforce the data point selection algorithm to choose at most half of the data points from the class of the original data point. Note that while data points are created by the genetic algorithm, the class labels are obtained by querying the black-box

model. Once they have the neighbourhood, they use the synthetic data points and train a decision tree. Finally, they take advantage of the decision tree and produce two types of rules; a single decision rule, and a set of counter-factual rules. A decision rule, generated from the path leading to the original instance in the decision tree, shows which attributes contributed the most to the decision made. Counter-factual rules, on the other hand, show alternative ways a user can change the decision by modifying the input attributes. These counter-factual rules are obtained by traversing all paths in subtrees originated from nodes along the original instance's path. Additionally, the leaf nodes in these paths must contain different labels than the label of the original instance to be considered among counter-factual rules.

3.6 Experiments with LIME

We conducted a few experiments on LIME to evaluate its performance with regard to certain parameters, and show the results here. The goal of these experiments is to see how trustworthy LIME's explanations are.

In the first experiment, we want to evaluate the quality of the explanations. As mentioned in the previous chapter (please refer to Section 2.4), we take advantage of a transparent classifier as a pseudo-black-box in our system to examine LIME's explanations. In all these experiments, we use different sample sizes to also find out more about the effect of sample size on explanations.

Our second experiment consists of modifying the number of buckets created during the discretization step. This step is needed to transform numerical data to categorical (multiple buckets) so that LIME can create randomly sampled data points that belong to different buckets.

3.6.1 Trustworthiness of Explanations

As we explained in the previous Chapter (Subsection 2.4) one way to evaluate a model-independent explanation framework such as LIME is to use a fully-transparent model instead of a black-box model in the experiments. In this

experiment, we use a decision tree — which we limit its depth to five [2]— as our black-box model. We want to understand the quality of the explanations produced by LIME when we increase the neighbourhood size from 1,000 to 10,000. In this experiment, we take advantage of four datasets from the UCI repository [20]. The results are presented in Figure 3.3 [3]. As it can be seen, while LIME tries to maintain a high recall score, it fails to provide analogous results in terms of precision. Precision here measures the proportion of the features identified as relevant by LIME, from the features actually used by the decision tree playing the role of the black-box. Recall, is the proportion of the features used by the black-box among the features indicated by LIME. Finally, the results vary greatly when dealing with different datasets (this is to some degree understandable since the number of the features varies across different datasets).

3.6.2 Effect of Different Bucket Construction Approach

In this experiment we show how the results differ when we use alternative bucket construction approaches. LIME uses buckets in order to convert numerical features from a continuous space to a discrete space. The methods include "quartiles", "deciles", and "entropy". In "quartile", LIME divides each feature space into four parts while in "decile" it divides it into 10 parts. "Entropy" exploits labels of instances to split the space according to the class labels. LIME package uses "quartile" as the default choice for discretization. Figure 3.4 shows the result of this experiment. It is evident that different discretization approaches leads to different results and there is no clear best method to use on a new dataset.

[2]So each explanation contains at most five important features.

[3]We further discuss the details of the datasets and experiments in Section 5.

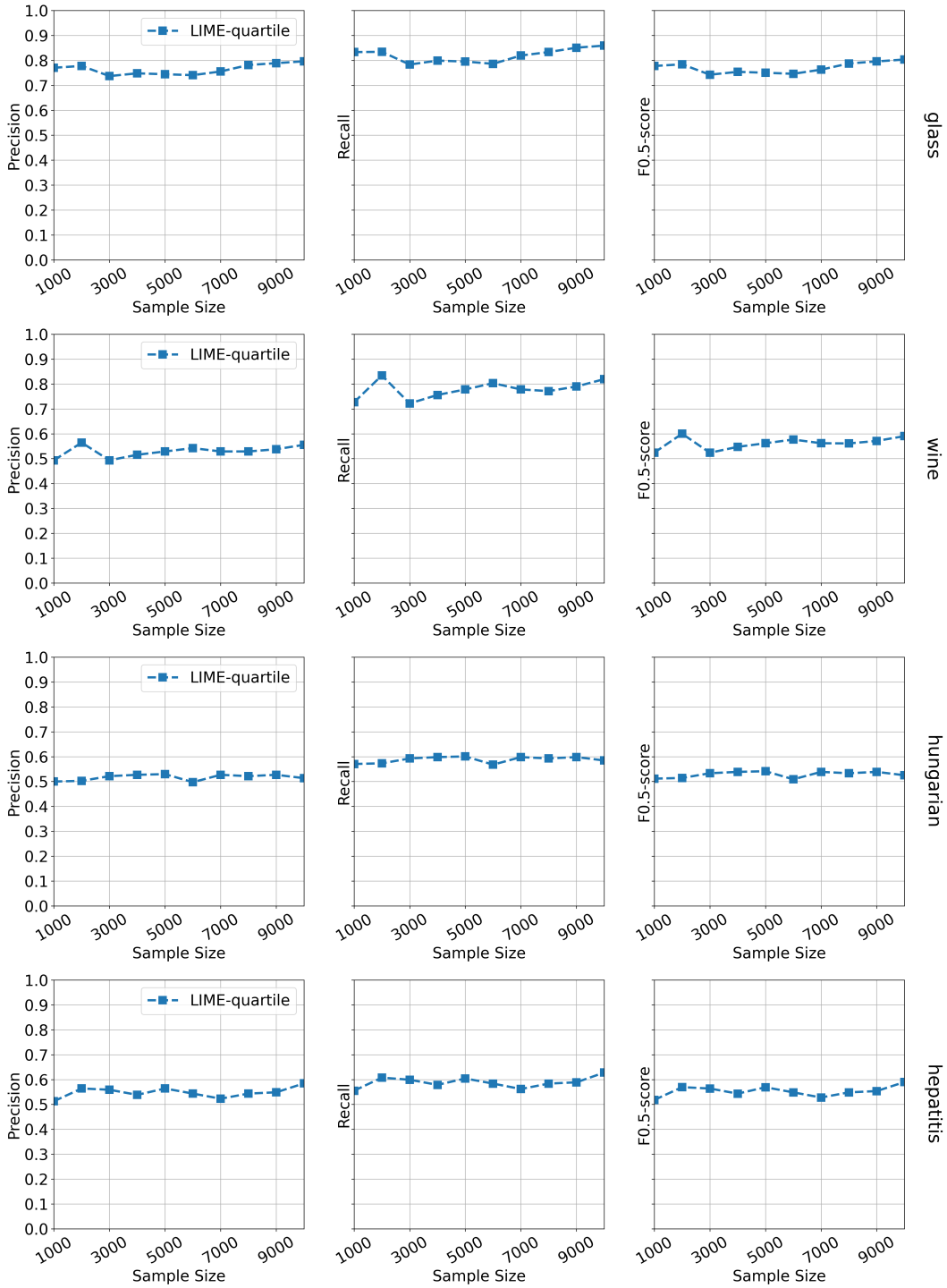

Figure 3.3: Precision, Recall, and F1-score for Glass, Wine, Hungarian, and Hepatitis datasets. The results extend from 1,000 synthetic points in the neighbourhood to 10,000.

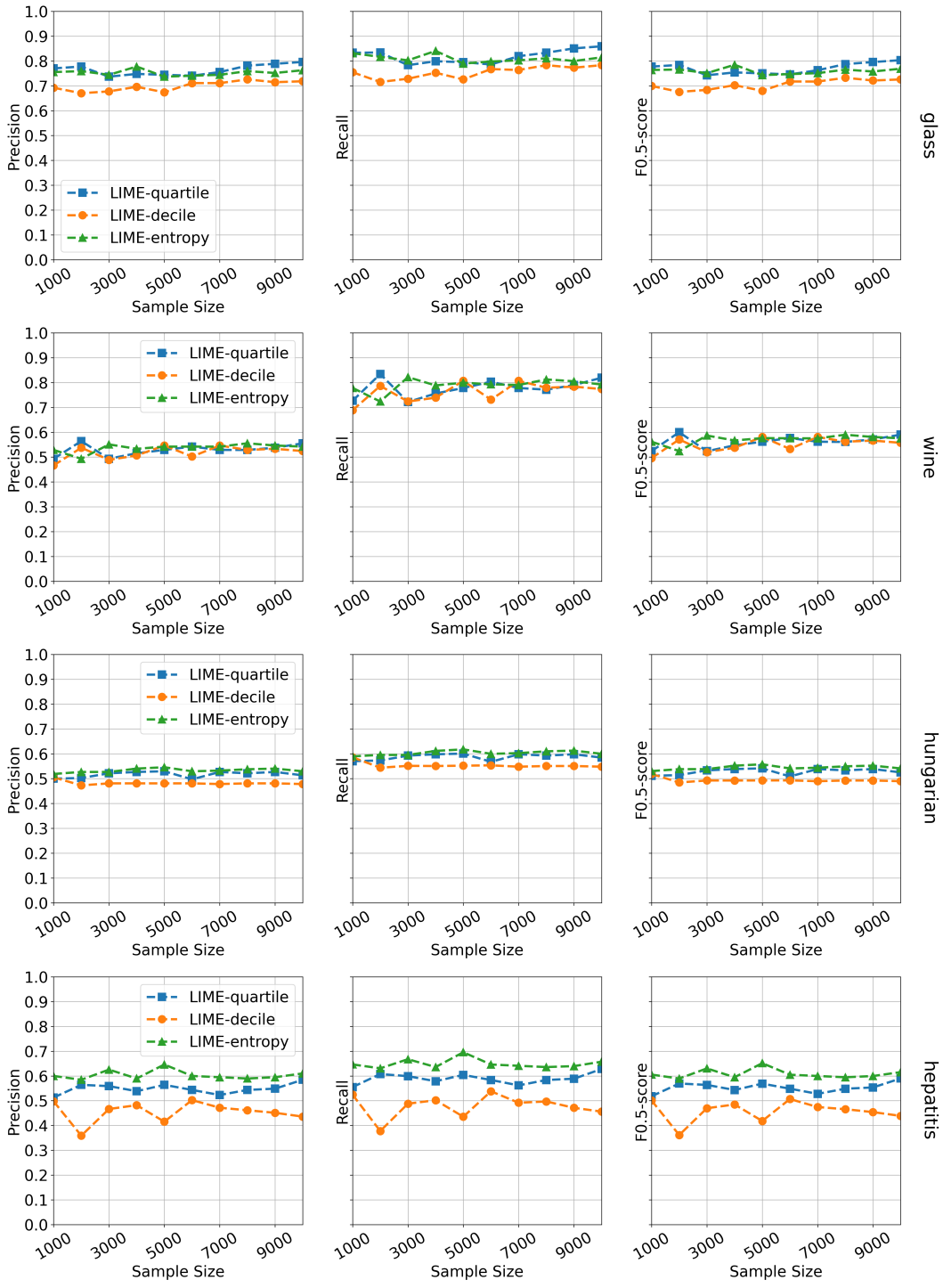

Figure 3.4: Precision, Recall, and F0.5-score for Glass, Wine, Hungarian, and Hepatitis datasets when using different bucketing approaches in LIME. The results extend from 1,000 synthetic points in the neighbourhood to 10,000. it is worth mentioning that there is no clear winner among all datasets.

Chapter 4

Association Rule-Based Classifiers

As mentioned in previous chapters, one important type of transparent classifiers is rule-based classifiers. At prediction time, these classifiers usually take advantage of a set of rules built at train time. Association rule classification is the product of applying pattern mining (mining frequent itemsets) in the classification task where, in short, frequent itemsets associated with a class label become rules representing that class label. The rules are conjunctions of feature-values implying a class label. f_1, and f_2, and f_3, and f_4, and ..., $f_n \rightarrow class1$

We first take a quick look at different rule-based classifiers in this chapter and then review the concept of association rules and association rule-based classifiers. We review one specific association rule classifier that we will use in our experiments, SigDirect, in more detail in the third section. Finally, we evaluate our implementation of SigDirect classifier using different datasets and compare its output against the numbers reported in the paper it was introduced.

4.1 Rule-based classifiers

Rule-based classifiers are, in principle, very similar to the "if" statements in computer programming languages, where a statement is executed only if a certain condition is met. Each rule in rule-based classifiers contains two

essential parts: a left-hand side (LHS) set of items and an item on the right-hand side (RHS), which should be the class label in the classification tasks. Alternatively, they can be called antecedent and consequent, respectively. In addition to these essential segments, different classifiers might append some further information to each rule, such as how frequently the LHS occurs in the dataset (support score), or how frequently the LHS co-occurs with the RHS in the dataset (confidence score).

Many different approaches have developed over the years that try to create rules in one way or another. In this section, we briefly review a few popular rule-based methods. The next section explains association rule-based classifiers.

4.1.1 OneR

OneR is a rule-based algorithm proposed by R. Holte [32] that essentially creates one rule for each value of each feature comprised of one feature in the antecedent and the label in the consequent. For each rule, the classifier stores the number of errors it made on the training set and leverages this at test time: the rule with the least error score among the applicable rules is chosen, and the label associated with it is returned as the final class label.

4.1.2 FOIL

Unlike OneR where each antecedent only contained one feature, First Order Inductive Learner (FOIL) [58] contains multiple features in the antecedent. This classifier, based on First-Order Logic, generates rules iteratively using the separate-and-conquer approach. In each iteration, through a top-down approach, a new rule is built by greedily adding new feature-values (e.g., color='red') to it. The choice of the next feature-value to be added to the rule is based on the Foil-gain metric, where the metric's goal is to contain as many positive examples as possible under the rule while limiting the negative examples. Once the new rule is built, all instances that the new rule could be applied to are removed from the current dataset, and the next iteration of the algorithm begins.

4.1.3 RIPPER

Another famous rule-based classifier that works on multi-class datasets is Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [13]. The classifier, similar to FOIL, uses the separate-and-conquer method to create new rules iteratively. This rule learner, however, splits the data into growing and pruning sets. Using FOIL's gain metric, it constructs a new rule based on the growing set and then exploits the pruning set to remove some of the feature-value literals added during construction — as long as the deletion helps create a better rule (using a heuristic metric that maximizes correctly classified cases while minimizing the number of incorrectly classified instances). The algorithm terminates when a rule is created that has an error rate above 50%. Notably, RIPPER introduces a post-processing step to further improve the quality of the rules created.

4.2 Association Rule-Based Classifiers

Association rules are one type of rules that take advantage of the associativity among the features. In fact, they are chiefly used in pattern mining, where frequent itemsets are extracted from transactional datasets. The Apriori algorithm [1], together with FP-Growth [29], are among the most prominent algorithms that extract such frequent patterns from datasets. In this section, after describing association rule mining, we review these two famous algorithms first, and then describe some methods that take advantage of them in the classification task.

4.2.1 Association Rule Mining

Association rule mining deals with finding an association between items in a transactional dataset. Given a dataset containing transactions that represent purchases in a grocery store, for example, the initial goal is to find frequent itemsets such as {'sugar', 'coffee', 'cream', 'milk'} and then, based on the frequency of each, generate association rules such as 'sugar', 'coffee' $\rightarrow$ 'milk', 'cream'. Unlike classification rules, there is no limit on the number and type

of items in the RHS itemset.

Apriori

The Apriori algorithm benefits from the idea that all subsets of a frequent itemset must also be frequent. It extends itemsets iteratively such that in each iteration, initially, all direct children of current itemsets are created by adding a new item to them should they be available in the train set, and then, removing the ones that are not frequent. To achieve this objective in a memory- and time-efficient manner, Apriori builds itemsets on a trie-like structure where the k^{th} layer holds all k-subsets of a feature set (except the ones whose ancestors had been discarded in earlier layers). Using this structure, each node in the trie represents a set of frequent items. This method allows us to easily generate all strong association rules from each remaining nodes in the trie.

FP-Growth

The Frequent Pattern Growth algorithm uses an extended trie-like structure called Frequent Pattern Tree (FP Tree) to compress the database into a more compact format. Once the FP Tree representing the dataset is built, the algorithm then creates further FP trees for each item. Finally, the algorithm recursively mines each of such trees to collect frequent itemsets.

4.2.2 Classifiers based on Association Rules

In this section, we review three association rule-based classifiers, and we take a more in-depth look at another one, SigDirect.

CBA

Classification Based on Associations (CBA) [41] is a classifier that extracts Classification Association Rules (CARs) from data in a fashion similar to the Apriori algorithm while limiting the rules to CARs. Unlike normal association rules where there is no limit on the number and the type of items in the consequent of a rule, the consequent of a CAR is limited to only one item,

which should be the class label. Once the algorithm has generated the rules, it first sorts the rules based on their confidence, support. Then, it uses the idea of database coverage and only keeps rules that cover at least one instance of the dataset if no preceding rule does so. They use the first applicable rule in the list mentioned above to classify unseen instances. There is also a default class (majority class) that the classifier applies if there is no applicable rule for the instance.

CMAR

Unlike CBA, which uses Apriori to generate the rules, Classification based on Multiple Association Rules (CMAR) [38] is another method that uses FP-Growth to generate rules. Once the algorithm generates all the rules, it uses an almost similar pruning approach as CBA. Finally, to classify each instance, the learner uses a weighted χ^2 measure to find the class label.

CPAR

Classification based on Predictive Association Rules (CPAR) [73] is a hybrid method that integrates traditional rule-based methods with association rule mining. This algorithm is based on the FOIL algorithm mentioned earlier, and uses FOIL gain for selecting the best feature in the algorithm. One main difference between CPAR and FOIL, however, is that CPAR assigns an importance score to each instance, and instead of removing samples covered by a generated rule, it decreases their importance by applying a weighted score. More importantly, instead of the greedy approach employed in FOIL, it keeps the top ones (close to the best one) when adding a new feature to a rule. Finally, it generates these rules simultaneously using association rule-based techniques; multiple features could be added in each iteration to the rule, thus making a trie-like structure.

4.2.3 ARC-AC and ARC-BC

Antonie and Zaiane introduce two different text classification algorithms in [3]. Initially, they introduce Association Rule-based Classifier with All Categories

(ARC-AC). Their method, similar to CBA, uses a constrained version of Apriori algorithm to create CARs. At prediction time, the algorithm first finds all applicable rules, but unlike CBA, ARC-AC computes the average confidence score for all different classes. The class with the highest average confidence score corresponds to the class label of the document.

Afterwards, they mention a shortcoming for methods such as ARC-AC that take advantage of frequent itemset mining: they are unable to properly find CARs in imbalanced datasets. To resolve this shortcoming, the authors propose Association Rule-based Classifier By Category (ARC-BC). This new classifier splits training documents into different categories where each category corresponds to one class. The algorithm then runs Apriori for each category similar to ARC-AC. This approach allows them to obtain frequent itemsets for different categories, thus making CARs for each class independently. This enables ARC-BC to classify imbalanced datasets very well. ARC-BC performs similar to ARC-AC at prediction time and picks the class label that has the highest average confidence score among all applicable rules.

4.3 SigDirect

We first review Statistically Significant Dependent Classification Association Rules for Classification (SigDirect) introduced by Li and Zaiane [37], which is a classifier that we use in our experiments we discuss in the next chapters.

We first explain how this classifier works by reviewing the statistical significance concept, and then close this chapter by comparing it against other classifiers mentioned earlier. We outline why we chose this classifier to explain decisions made by black-box systems.

4.3.1 Statistical Significance

Unlike other methods that use minimum thresholds for support and confidence values to discard unreliable rules, SigDirect benefits from the Statistical Significance concept.

In order to show that the result produced in an experiment is not due to

errors such as sampling error, we can apply the Statistical Significance test. In the Statistical Significance test, assuming a null hypothesis (i.e., having no relationship between measured variables, and an outcome occurring only due to chance), we want to show that such a hypothesis is improbable.

More formally, given a threshold α, chosen based on the field of study, the result of an experiment is said to be statistically significant if the p-value (the probability value of getting results as extreme as in a null hypothesis) is less than α. The α is usually set to 0.05 for most scientific research.

The formula to compute the p-value for a classification association rule given in the form of $X \rightarrow c_k$ can be computed using the Fisher's exact test:

$$p(X \rightarrow c_k) = \sum_{i=0}^{min\{\sigma(X,\neg c_k),\sigma(\neg X,c_k)\}} \frac{\binom{\sigma(X)}{\sigma(X,c_k)+i}\binom{\sigma(\neg X)}{\sigma(\neg X,\neg c_k)+i}}{\binom{|D|}{\sigma(c_k)}} \tag{4.1}$$

As one can see, computing such value is very expensive. However, the author of [28] introduced a formula to efficiently compute the lower bound for p-value when $\sigma(c_k) \geq \sigma(X)$:

$$p(X \rightarrow c_k) \geq \frac{\sigma(\neg X)!\sigma(c_k)!}{|D|!(\sigma(c_k) - \sigma(X))!} \tag{4.2}$$

Using this formula, one can compute the lower bound for the rule, and unless the lower bound is less than α, there is no need to compute the exact p-value anymore since the rule cannot be statistically significant.

4.3.2 SigDirect Details

SigDirect uses an Apriori-like strategy to first generate the rules and then leverages an instance-based approach for the pruning step to only keep rules with the highest quality and discard the rest. Similar to Apriori, it expands the k^{th} level to build the $k+1^{th}$ level using the train set. In the k^{th} iteration, for example, each node is evaluated in this way: initially, Equation 4.2 is used to compute the lower-bound of p-value and then if the lower-bound is below the statistical significance threshold set by the user, Equation 4.1 is applied to get the exact p-value. This method helps avoid the cost of computing the exact p-value (which is expensive) for rules that we already know are not significant.

Once the algorithm computes the p-value, it then evaluates the minimality and non-redundancy of the candidate rule to determine if it could be selected. The p-value associated with all parents of the LHS of the candid rule (i.e., all $n-1$ subsets of an itemset with n features) should not be less than the p-value of the candidate rule. Finally, to determine if a candidate rule is minimal, we make sure all occurrences of such LHS itemsets in the train set belong to the same class.

4.3.3 Advantages over other classifiers

One of the main advantages of using SigDirect as the transparent model over other classifiers such as CPAR or CMAR is the fact that we do not need to tune any hyper-parameter in SigDirect; support and confidence thresholds play a critical role in CBA and CMAR. This advantage comes in handy especially in explaining a black-box model where we need to train the explainer (i.e., the transparent model such as SigDirect) for every individual decision of the black-box (i.e., different test instances) is essential. Moreover, there is generally no ground truth data available to evaluate the quality of the explanations created by the explainer model. Thus, there is no easy way at all for the end-user to tune any hyper-parameters.

Also, Li and Zaiane [37] showed that not only does SigDirect generally outperform other classifiers in accurately classifying instances of different datasets, but it also creates fewer rules than them. This advantage allows the end-user to more easily understand the explanation, hence making it an appropriate classifier for this task.

4.4 SigDirect Implementation

Here, we provide details about our implementation of SigDirect. The implementation is in Python 3 programming language, and takes advantage of some of the numerical libraries in Python such as Numpy and Scipy in order to do the computation faster than pure Python code[1]. The reason we used Python

[1]You can access the code at `https://github.com/mhmotallebi/sigdirect`.

— and not other programming languages such as C++ that are generally faster — is that it allows other researchers to further improve SigDirect or modify it according to their needs. This is important since Python has become the leading programming language used in ML, both in academia and industry where research has shown that 57% of the machine learning developers and data scientists use it [72].

4.4.1 Evaluation

In the table below, we compare our implementation of SigDirect against the results provided in the paper introducing the algorithm [37]. We used 17 datasets of the UCI repository [20] and provide the accuracy when using different rule selection heuristics at prediction time. We also show the number of rules generated after the training time, and also the number of rules that remained after the pruning step. All numbers are results of 10-fold cross-validation runs [2,3].

[2]Splits were identical to the ones used in the original paper.

[3]We also contacted the author of SigDirect paper to obtain their code. We received an executable file. When we compared our results, the numbers produced by the executable and the Python implementation were closer than that of the paper.

	Reported in Paper					Python Implementation				
dataset	**S1**	**S2**	**S3**	**I-Rules**	**F-Rules**	**S1**	**S2**	**S3**	**I-Rules**	**F-Rules**
Adult	84.0	83.9	84.0	136.1	50.1	83.1	82.2	82.1	121.2	77.7
Anneal	92.1	91.6	93.5	340.5	23.2	96.9	94.1	96.8	372.3	42.8
Breast	91.4	91.7	91.6	19.8	10.1	91.4	91.7	91.6	25.1	10.9
Flare	80.6	81.8	81.6	600.9	45.1	83.1	84.2	84.3	635.3	75.8
Glass	66.3	69.2	68.7	340.5	23.2	66.4	70.1	67.8	279.4	56.2
Heart	55.8	58.1	57.4	19.8	10.1	56.4	58.7	57.4	520.2	79.3
Hepatitis	83.2	85.2	82.6	185.7	19.0	84.5	85.8	84.5	196.1	33.3
HorseColic	81.0	80.2	81.0	319.9	33.3	81.3	81.3	81.3	325.1	89.1
Iris	89.3	89.3	89.3	8.1	6.0	94	94	93.3	14.1	6.2
Led7	73.5	73.4	73.6	269.0	108.7	73.5	73.7	73.4	341.2	104.9
LetRecog	48.0	58.8	52.4	19252.9	468.0	46.2	56.5	50.7	42301	2472
Mushroom	100.0	100.0	100.0	906.4	16.0	100	100	100	865.1	106.4
PageBlock	91.2	91.2	91.2	230.1	25.8	91.1	91.2	91.2	223.1	30.1
PenDigits	84.3	88.4	84.6	8406	212.0	76.5	82.4	77.8	8381.9	759.1
Pima	74.6	75.1	74.6	116.8	33.2	74.7	75.3	74.7	124	36.7
Wine	92.7	93.3	92.7	8721.7	11.1	92.7	92.1	92.7	108.9	29.8
Zoo	94.1	94.1	94.1	4597	7.8	92.1	93.1	93.1	2282.8	14.7

Table 4.1: Evaluation of Python Implementation of SigDirect classifier against the numbers reported in the paper [37]. Each row corresponds to one dataset. S1, S2, S3 are different heuristics used at prediction time where S1 uses p-values of applicable rules to decide the final class label. Confidence score of rules and the multiplication of it by p-values are used in S2 and S3 heuristics, respectively. I-Rules refers to the initial rules generated by the classifier while F-Rules shows the number of final rules, the remaining rules after pruning the initial rules.

Chapter 5

BARBE: Black-box Association Rule-Based Explanations

In this chapter, we introduce Black-box Association-Rule Based Explanations (BARBE). It is a method that takes advantage of association rules, more specifically associative classification rules, to provide better and more human-understandable explanations compared to other methods such as LIME.

The breakdown of this chapter is as follows: first, we mention a challenge that exists in LIME and other model-agnostic explainers in Section 5.1. Then, we introduce BARBE and explain how it overcomes that shortcoming in Section 5.2.

Section 5.3 provides an in-depth description of the constructing components of BARBE. In this section, we also elaborate on what the role of each component in BARBE is and how they contribute to the framework.

We review our experiment settings in Section 5.4. We first mention the datasets we used in our experiments and discuss the methodology and the corresponding context.

Rank-based comparison is the topic of Section 5.5 where we mention why it is an important part of our experiments.

In Section 5.6, we first discuss and then evaluate different settings[1] available in BARBE and the need to find the best choice for. In this section, we find the best settings for BARBE that work best among different datasets.

[1]Please note that while a few of these may be considered hyper-parameters, the rest of them are not.

We compare the performance of BARBE against LIME in Section 5.7 where we want to know how BARBE's explanations contrast against a well-known model-independent framework.

Finally in Section 5.8, we provide some future work suggestions to continue and improve BARBE.

5.1 A Big Challenge with XAI: What Counts as Explanation

There is, unfortunately, no agreement among researchers on what counts as an explanation and what does not. Therefore, different researchers have considered a variety of outputs as an explanation. There are many research methods that, for example, generate a sentence explaining the outcome of a model classifying images while some other work use heat-maps to provide explanations.

Take what LIME generates for tabular datasets as another example (see Figure 3.2). What do the numbers in front of each feature mean to you? What is the difference of *population=solitary* having the importance score of 0.12 and *stalk−root=bulbous* weighing at 0.10? The only conclusion we can have is the fact that *population=solitary* has a higher importance in this instance than *stalk−root=bulbous*. This leads us to the conclusion that only the order among features matters to the users and not the numbers generated in the explanations.

One may suggest that these numbers can help us understand how to alter the class label. They may, for example, argue that if the sum of two features is more than that of a third feature, then if we change these values, it could lead to a change in the class label (of course this only makes sense in binary datasets).[2] While this is what the LIME authors claim, the truth is that since they use a weighted loss function for their linear model that also benefits from regularisation, it likely that these points which are not in the very near proximity of the original data point would be misclassified by this linear model, thus providing wrong explanations to the user.

[2]This is mentioned by LIME authors

5.1.1 Rules: A Better Way to Explain

Ribeiro et al. [60], the same authors of LIME, also point out another shortcoming of methods like LIME (please refer to the example shown in Figure 2.4). They introduce Anchor to overcome this issue. In their new method, an explanation is a set of features that whenever they co-occur, the class label is determined with a 95% confidence. This Anchor essentially resembles a rule (with a high confidence threshold of 95%).

The authors of LORE [25] benefit from the idea of using a rule as the explanation in their method as well. In their method, however, Guidotti et al. also provide a set of counter-factual rules helping the user find ways to have a new data point which is labelled differently than the original one while having the least different features compared to it.

Despite the fact that these methods, to some degree, overcome the problem mentioned above, one issue still remains: is there always only one set of correlative features (and hence one reason) behind the final outcome of the model? What if there were multiple sets of correlative features that independently derive the final conclusion of the system [45].

5.2 A Solution to the Challenge: BARBE

To overcome the shortcomings mentioned above, we introduce Black-box Association Rule-Based Explanations or BARBE.

Our method, unlike LIME, provides a set of rules as the explanation, where not only do rules provide users with important features (what LIME does), but also takes care of the associations among them (what LORE and Anchor do). In addition, since we provide multiple rules as an explanation, we can hint on multiple causes that have led to that decision by the system, something that the aforementioned methods are unable to provide. Note that using a decision tree which can be converted into a set of rules, the path in the tree leading to the predicted label constitutes a one unique explanation.

Feature Number	Feature Name	Bucket Number
1	Refractive Index	7
2	Sodium	3
3	Magnesium	1
4	Aluminum	3
5	Silicon	0
6	Potassium	1
7	Calcium	3
8	Barriom	0
9	Iron	3

Figure 5.1: An instance of Glass dataset with nine features. The number in front of each feature is the bucket number for the value of that feature in this instance.

5.2.1 What does BARBE's explanations Look Like?

In BARBE, a set of rules is returned as the Explanation. In addition, it is capable of providing an ordered set of important features as an alternative way to provide explanations. This allows the users to have the choice to look at these two types and get a better understanding about the explanation. Each rule, in addition to the items in its antecedent (or LHS) and the class label, contains other information such as its confidence, support value, and the p-value. These statistics can further contribute to the information the user could infer from the black-box model. Ultimately, a set of features sorted based on their importance is returned. This not only makes an alternative way for the user to grasp the black-box model, but it also paves the way for us (i.e., BARBE developers) to make comparisons against other methods such as LIME.

Figure 5.2 contains an example of what BARBE outputs for the instance shown in Figure 5.1. In this example, BARBE produces three rules in which they not only provide important features to the users, but they also hint on the associations among the features, further helping the user understand the way the black-box model has to come to its decision.

RULES	*Support*	*Confidence*	*ln pF*
3="1" AND 4="3" AND 7="3" → 3	0.0900	0.833	-17.095
3="1" AND 6="1" → 3	0.2230	0.791	-33.641
6="1" → 3	0.3730	0.696	-24.174

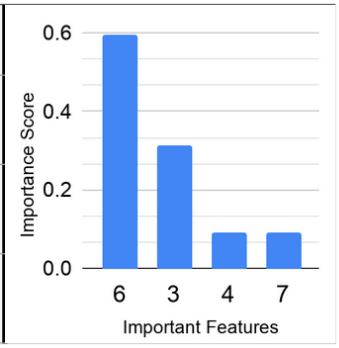

Figure 5.2: The explanation provided by BARBE for an instance of the Glass dataset is shown in Figure 5.1. Here, feature numbers are shown for conciseness. The right side contains the important features ranked based on their importance. The left side of the figure contains important rules. Each rule includes LHS items together with the label. Besides, support score, confidence score, and the logarithm of the p-value reported by SigDirect are also presented to the user.

5.3 How does BARBE work?

A high-level representation of BARBE's activity diagram is shown in Figure 5.3.

Figure 5.4 further illustrates the architecture of BARBE in more details. BARBE contains two parts: 1) one part required for the whole dataset, and 2) another part processed for each instance of the dataset that requires explanation.

Initially, BARBE takes advantage of the training data and creates multiple buckets for each feature. This allows BARBE to discretize continuous features and replace each value with the corresponding bucket number. Using this approach, BARBE can replace all numerical features with ordinal ones.

Every time the user requests an explanation for an instance of interest, BARBE initially creates random data points in the vicinity of the instance. Once BARBE created these data points, it then "undiscretizes" them and sends them to the black-box classifier for labelling. The "undiscretization" step transforms features — that had a continuous space in the original data point, but BARBE had transformed them into buckets in the discretization step — back to their original form. To make such a transformation, it creates

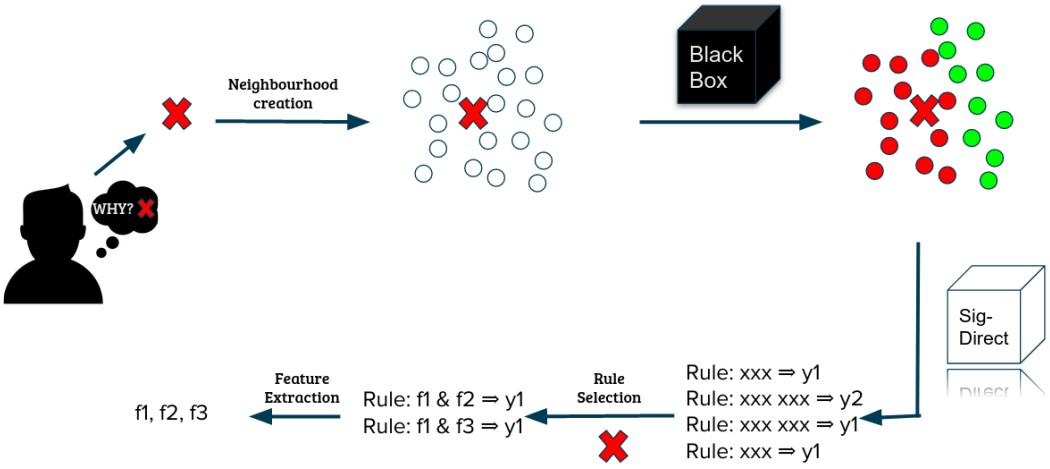

Figure 5.3: A simplified version of how BARBE produces explanations for an instance. Initially, BARBE creates a neighbourhood around the data point containing synthetic data points. Afterwards, BARBE queries the black-box to label instances in the neighbourhood. These synthetic data points and their corresponding labels are then sent to SigDirect classifier to train a supervised model. The outcome of training this model is a set of rules. The original data point is then used to extract relevant rules from the trained model. Lastly, BARBE extracts important features from these rules and reports them, together with the rules, as the explanation to the user.

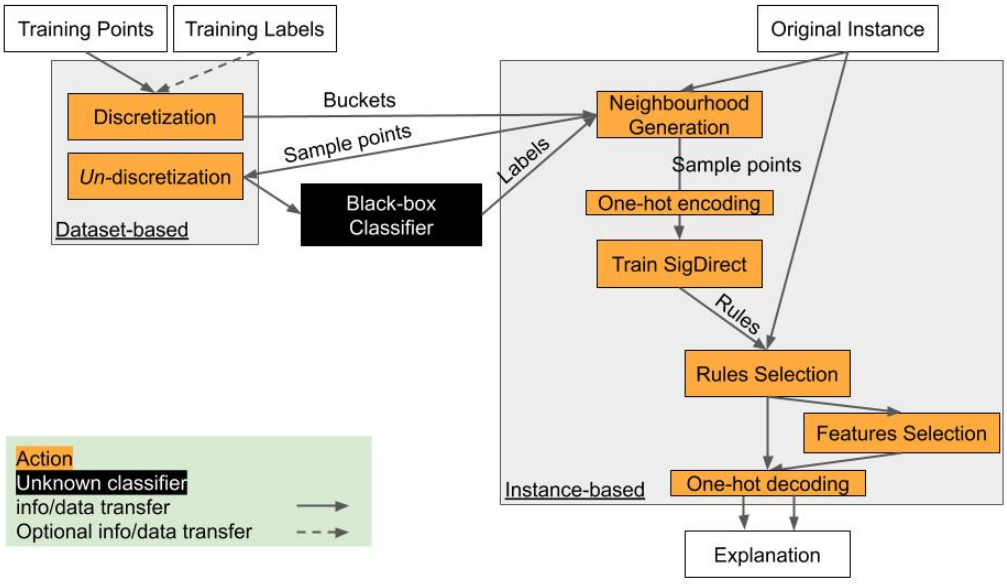

Figure 5.4: BARBE's architecture, and how it produces an explanation for the requested instance. This figure exhibits the different constructing components of BARBE in detail. The grey box on the left side of the figure shows the part that needs to be created for a dataset while the one on the right side needs to be produced for each data point in which the system is queried for an explanation.

a normal distribution for each bucket of a feature and then randomly samples from the distribution.

This combination becomes a new training dataset for BARBE. Afterwards, BARBE performs one-hot encoding on them to transform them into *transactions*, the suitable data format for SigDirect, the interpretable classifier BARBE uses under the hood. It next trains the associative classifier using this dataset, which produces a set of rules.

While these association rules are all generally relevant to the data point — since the training data is made of instances from the vicinity of the original data point — the next step further narrows down the most relevant rules. As mentioned in the previous section, BARBE is also capable of providing the most important features. In the final step where it selects the most relevant features from the important rules. Eventually, it decodes all the features in the selected rules and features into the feature space of the black-box (i.e., one-hot decode them).

5.3.1 Discretization

The discretization step, which is based on the same step in LIME, splits continuous features into discrete ones. Our current implementation supports dividing continuous features into 1) quartiles, 2) deciles, or 3) multiple buckets according to the entropy. When splitting according to the quartiles method, we split the space of each feature into four equi-depth buckets. In decile mode, the number of buckets increases to 10. There is no need to have access to the training data in the first two, and only the distributions of values for each feature suffice. The entropy-based method, however, requires the training data, including the labels, to break the feature space into meaningful buckets.

5.3.2 Neighbourhood Generation

One important step that many model-independent methods have in common is the neighbourhood generation process (That is why they are sometimes called perturbation-based methods). This step, together with the next, are the fundamental parts of all model-independent frameworks. Most of these

methods, have to rely on either creating data points in the vicinity of the instance or assigning a higher weight to the points nearby. Unlike LIME that benefits from both techniques simultaneously, BARBE only relies on the first one. This is done for each feature by randomly creating a normal distribution around the original bucket. For nominal features, we use the frequency of each value in the training data to create the distribution, similar to the approach adopted in LIME.

5.3.3 Interpretable model (SigDirect)

Once the discretized data points are created, they are first undiscretized based on the bucket's information (so the black-box model could be able to handle them), and then sent to the black-box model to be labelled. By one-hot encoding the discretized data, and combining it with their corresponding labels, the synthetic training data is ready to be fed to SigDirect. At this point, SigDirect is trained and a set of rules is returned. It is worth noting that this is different than what LIME does. In LIME, the undiscretized data is fed to the interpretable model (linear regression) while we take advantage of discretized data in BARBE. Besides, LIME processes and relies on the probability scores returned by the black-box model while we only benefit from labels.[3]

5.3.4 Rule Extraction

This is the step where given the trained interpretable model, BARBE extracts pertinent rules from the set of rules that SigDirect had generated at train time. This is achieved by exploiting "applied" and "applicable" rules. "Applicable" rules are the rules that include the same feature-values as the original instance while "applied" rules are a subset of applicable rules that their labels must agree with the instance label from the black-box. Equation 5.1 shows an instance of a toy dataset aiming at classifying cars as expensive or non-expensive. This equation describes a white Porsche sports car built between

[3]This is, in fact, one of the advantages of BARBE that it only requires labels from black-box models and, unlike LIME, can explain black-box models that do not provide such probability scores.

2010 and 2015.

$$colour{=}white \ \& \ build{=}[2010{-}2015] \ \& \ type{=}sport \ \& \ brand{=}Porsche \tag{5.1}$$

What follows are a set of rules that could be created by a rule-based classifier. In these rules, Rules 5.2, 5.3, and 5.4 are considered "applicable" since they all contain the same feature-values as the instance we showed earlier. Rules 5.2, and 5.3 are also "applied" since they agree on the class label as well. This is not true for Rule 5.4 since it does not agree on the class label. Finally, the rest of the rules such as the ones in Equations 5.5, and 5.6 are neither applied nor applicable rules for this instance because of the colour features and the type feature in 5.6.

$$brand{=}Porsche \implies price \geq \$30k \tag{5.2}$$

$$type{=}sport \ \& \ brand{=}Porsche \implies price \geq \$30k \tag{5.3}$$

$$colour{=}white \ \& \ build{=}[2010{-}2015] \implies price < \$30k \tag{5.4}$$

$$colour{=}red \ \& \ type{=}sport \implies price \geq \$30k \tag{5.5}$$

$$colour{=}brown \ \& \ type{=}sedan \implies price < \$30k \tag{5.6}$$

5.3.5 Feature Extraction

Lastly, BARBE exploits important rules to extract important features. In this step, the user can optionally provide the system with a number k and the system provides up to k important features accordingly. Otherwise, BARBE provides all the important features it has found in the important rules. To get these features, BARBE looks at the rules it extracted in the previous step and ranks them according to the information provided by associative classifier (i.e., confidence, support, and p-value scores).

dataset	# train	# test	# features	# classes	avg explanation size[5]
Glass	160	54	9	6	4.41
Wine	133	45	13	3	3.43
Hungarian	220	74	13	2	4.35
Poker	25,010	1,000,000	9	4	4.86
Breast	524	175	9	2	3.36
Image	210	2,100	19	7	3.45
Magic	14265	4,755	10	2	3.75
Vowel	742	248	13	11	2.86
Hepatitis	116	39	19	2	4.45
WPBC	148	50	33	2	3.6
WDBC	426	143	30	2	4.26

Table 5.1: Datasets used in the experiments, and some information regarding them.

5.4 Experiments on BARBE

In this section, we discuss the settings under which we conduct many experiments to evaluate BARBE in the later sections. We first introduce the datasets we used and then discuss the approach we took to evaluate BARBE. Later, we provide the metrics that we benefited from in our experiments. Finally, we discuss how fidelity plays a vital role in BARBE.

5.4.1 Datasets

Since we are interested in creating a framework that could be used on any tabular dataset, we need to fine-tune BARBE on multiple datasets to make sure it would perform robustly on unseen ones. To achieve this, we leverage the UCI repository [20]. We select various datasets from this repository to conduct experiments. Table 5.1 lists the datasets we use in our experiments and provides some insights concerning them. We use the train sets[4] of these datasets to train a decision tree representing the black-box in our experiments. Also, we use the train sets of the first 9 datasets in the provided list to tune BARBE (Section 5.6) while we take advantage of the test sets of all datasets to compare against the other method (Section 5.7).

[4]If the dataset is not split, we randomly select 75% for training and 25% for test.

5.4.2 Experiments Setup

As we mentioned in Section 2.4, evaluation is a big challenge in the XAI literature. We use the method described in Section 2.4 for model-independent explainers where we replace the black-box model with a fully-transparent one. This allows us to get the explanation from the black-box model which corresponds to the ground truth data required for our experiments.

The interpretable model we leverage in our experiments is a Decision Tree (DT)[6,7]. As we have discussed in previous chapters (see Section 2.4.3), we can extract the explanation from a DT by retrieving the features present in the path the inference algorithm uses to infer the instance's class label.

We limit the depth of the DT to a specific number k at train time. In such a way, we guarantee that the explanation comprises of at most k features. Note that this does not mean there are exactly k features present in all paths in the DT. This, however, is used to merely limit the depth of the DT. Hence, we do observe trees that use as few as one feature to label some instances, making the explanation very short.

5.4.3 Experiments' Metrics

We used *Precision*, *Recall*, and F_β*-score* as our metrics in the experiments. We adopted *Precision* since it helps us understand the correctness of individual features present in the explanations. In other words, it measures how many of the features BARBE has said important are important. On the other hand, *Recall* is critical to help us comprehend the coverage of the true explanation by the one produced by BARBE. It informs us how many of the important features are retrieved by BARBE. We take advantage of F_β*-score* to combine the two metrics mentioned above. In this metric, β is decided by the user and implies the relative importance of *Recall* over *Precision*. An F_β*-score* with $\beta = 2$ means *Recall* is twice as important as *Precision*. Although F_1*-score* is a harmonic mean of the two metrics mentioned earlier and considers them

[6]Please note that based on our discussion in Section 2.2.1 the depth of the tree should be limited to a reasonable number to be fully-interpretable.

[7]We use scikit-learn [55] for implementing the DT.

equally important, we utilised $F_{0.5}$-*score* in our experiments. That is because we believe *Precision* is more important than *Recall*: if an explanation provided to the end-user suggests most features are salient (i.e., a case where *Precision* is low but *Recall* is very high), the end-user is not able to trust the system since that explanation is not insightful. If the explanation, however, includes only a few of the features which are mostly tagged correctly as important (i.e., a case where *Precision* is high but *Recall* is low), then the end-user can trust the system more as this case indicates the black-box model is focusing on the right features.

In our experiments, we compared the explanation provided by BARBE with the ground truth explanation obtained from the DT for each instance in the dataset. We then calculated the metrics for each one. These metrics are later averaged over all instances used in the experiment and finally reported in the figures. The figures in each experiment only include three datasets with the most observable differences, while the results for other datasets are reported in the corresponding sections in Appendix A.

We restricted the number of instances we evaluated to only 100 from any given dataset to speed up the evaluation. Although we randomly selected these instances, they were identical in all experiments.

5.4.4 Fidelity to Global Model

We want to understand if the interpretable model in BARBE (i.e., SigDirect) can correctly classify the instance or not. That is, we query the associative classifier to see if its predicted class label matches the class label predicted by the black-box model.[8]

BARBE provides an explanation only if the labels agree. Otherwise, it does not produce any explanation. Besides, for tuning purposes, the numbers we report for each instance in Section 5.6 (*Precision*, *Recall*, and $F_{0.5}$-*score*) are set to zero if the labels do not agree. This is because we want to tune

[8]Please note that the true label is what the black-box has predicted and not what the dataset provides since BARBE is explaining the outcome of the black-box classifier and not the real cause behind an event, which can be different than what the classifier has learned.

BARBE to jointly maximize the metrics mentioned earlier, and the fidelity score. In Section 5.7, however, we provide fidelity as a separate metric and provide the average *Precision*, *Recall*, and $F_{0.5}$-*score* for instances that have satisfied the fidelity criterion.

5.5 Rank-based Comparison

Rank-based comparison is another metric we leverage in a few of our experiments. In this section, we initially explain why we need it and then review the proper metric for it.

5.5.1 Why Rank-based Comparison?

While using *Precision*, *Recall*, and $F_{0.5}$-*score* helps us quantitatively assess the explanations, they are not adequate. Many times, although having a set of features as the explanation is significant, there is a difference between the first and the last one in terms of their relative importance. For example, if features A, B, and C are the important features in a data point (with the same order), an explanation framework outputting them with the same order shall be given more credit than another framework returning C, B, and A as the explanation. To consider this fact, we want to investigate their rank within the important features. In this way, an explanation framework that puts the most important feature on top of the list would be distinguished from another one that barely includes it in the explanation.

5.5.2 Rank-based Comparison Alternatives

One notable measure for comparing two lists based on their ranks is the Spearman's Rank Correlation Coefficient (ρ measure). This measure takes into account the order of the features and then assigns a similarity score between -1 and $+1$. The two extremes occur only if the two lists perfectly match. Three main disadvantages of this measure for our task, however, are 1) this measure does not work when we have uneven lists (i.e., the size of the explanation provided by BARBE is different than the ground truth explanation),

2) the same ranks should be present in the lists, and 3) it does not differentiate features' importance based on their rank in the lists (i.e., it is not a weighted comparison). To resolve these shortcomings, we explored alternative metrics developed in similar tasks. One particular research field that requires many such evaluations is Information Retrieval (IR). A common task in IR comprises evaluating a given set of documents: when a user queries for a topic, the search engine returns the most relevant documents. Initially, the search engine assigns a similarity score (with respect to the topic) to each document and then returns them according to their similarity score (the higher the similarity score the earlier it is shown to the user). How do IR researchers know if their system retrieves the relevant documents properly? Assuming they have a ground truth order for documents, they should compare their list against that ground truth to see if their system is working properly. A mismatch at the beginning of the list must have a higher penalty than a mismatch at the end of the list. This scenario is identical to our case where our explanation — which in our case is a list of features — should mirror the ground truth.

One of the famous measures used in IR is Rank-Biased Overlap (RBO) [69]. This measure exploits the overlap between the two lists. One characteristic that distinguishes this metric from others such as Spearman's ρ is that this is a weighted measure. Another characteristic is that it works even if some features are present in only one ranking. Finally, it is capable of working on uneven lists.

Equation 5.7 presents the RBO measure. This similarity measure assigns a score between 0 and 1 when comparing two lists. In this equation, L and S refer to the longer and shorter lists, respectively. Additionally, s refers to the length of S while l refers to the length of L. X shows the length of the overlap between the two lists, and p is a parameter set by the user to assign the importance of top features.[9]

[9]We used the implementation provided in `https://github.com/changyaochen/rbo` in our experiments.

$$RBO(L,S,l,s) = \frac{1-p}{p}(\sum_{d=1}^{l}\frac{X_d}{d}p^d + \sum_{d=s+1}^{l}\frac{X_s(d-s)}{s\times d}p^d) + (\frac{X_l - X_s}{l} + \frac{X_s}{s})p^l \tag{5.7}$$

5.6 Tuning BARBE

We explore several BARBE settings, such as hyperparameter values, in this section. To discover the best setting, we run experiments on different datasets to determine the setting that outperforms the rest. This allows us to have a robust framework in which the end-users does not need to tune hyperparameters anymore, and can use BARBE off-the-shelf with almost no need to change anything.

As mentioned in previous sections, many settings in BARBE allow us to modify its architecture such as minor modifications to SigDirect (e.g., setting the p-value threshold, or adding the early-stopping feature to it) or the way we create the neighbourhood around the instance. We choose to start with the parameters in SigDirect and find the best choice among them. We initially explore the effect of different p-value thresholds in SigDirect and then examine the impact adding early-stopping causes to SigDirect. Next, we evaluate the effect of different sampling coefficients in BARBE among different datasets. Afterwards, we review the impact of having a multi-class dataset compared to a binarized one for multi-class datasets. The next experiment looks into different rule selection strategies in BARBE. Another experiment we conduct is in regards to improving the fidelity of BARBE. We conclude these experiments by analyzing how taking advantage of different feature extraction methods impact the outcome.

Here are the initial settings we use in our experiments: We first use early-stopping in SigDirect in our experiment since it allows BARBE to create explanations faster. We also use binary datasets as these are the settings that were used in LIME. Furthermore, we take advantage of applied rules followed by other applicable rules in the feature extraction step.

As we mentioned earlier, we take advantage of the training sets in these experiments to tune BARBE (however, when comparing to other explanation techniques, we use test sets of the datasets as our hold-out sets).

Another crucial point we should emphasise concerning the experiments is that BARBE outputs an explanation only if its transparent model (SigDirect) and the black-box model agree on the class label of the instance. In other words, BARBE respects the fidelity to the model and if the labels do not agree, BARBE provides no explanation.

5.6.1 P-value Threshold in SigDirect

The only hyperparameter in SigDirect classifier [37] is the threshold set for p-value so that if the p-value of a rule is less than the threshold, it would be considered as a statistically significant rule. The authors of the paper set this threshold to 0.05 since they claim this is the value commonly used in scientific experiments. Nevertheless, we were interested to examine the consequence of different thresholds in BARBE. If we decrease this threshold, rules created by SigDirect would be more statistically significant, and also fewer of them would be created. Since we are only interested in rules regarding one data point, we believe these advantages can be very helpful for our explanations. To verify this hypothesis, we conduct an experiment where the goal is to see if decreasing the p-value threshold in SigDirect results in better explanations from BARBE. We report the results in Figure 5.5 where we show that by decreasing the p-value threshold, we gain more statistically significant rules which eventually prompt a higher *Precision* among different datasets. A side effect, however, is that it leads to creating fewer rules and thus having lower *Recall* scores.

5.6.2 Early-Stopping in SigDirect

We noticed we could stop the SigDirect tree generation algorithm when it did not add any rule after expanding a new layer in the tree[10]. We can terminate

[10]Note that we need a statistically significant node — together with some other conditions met — to generate a new rule.

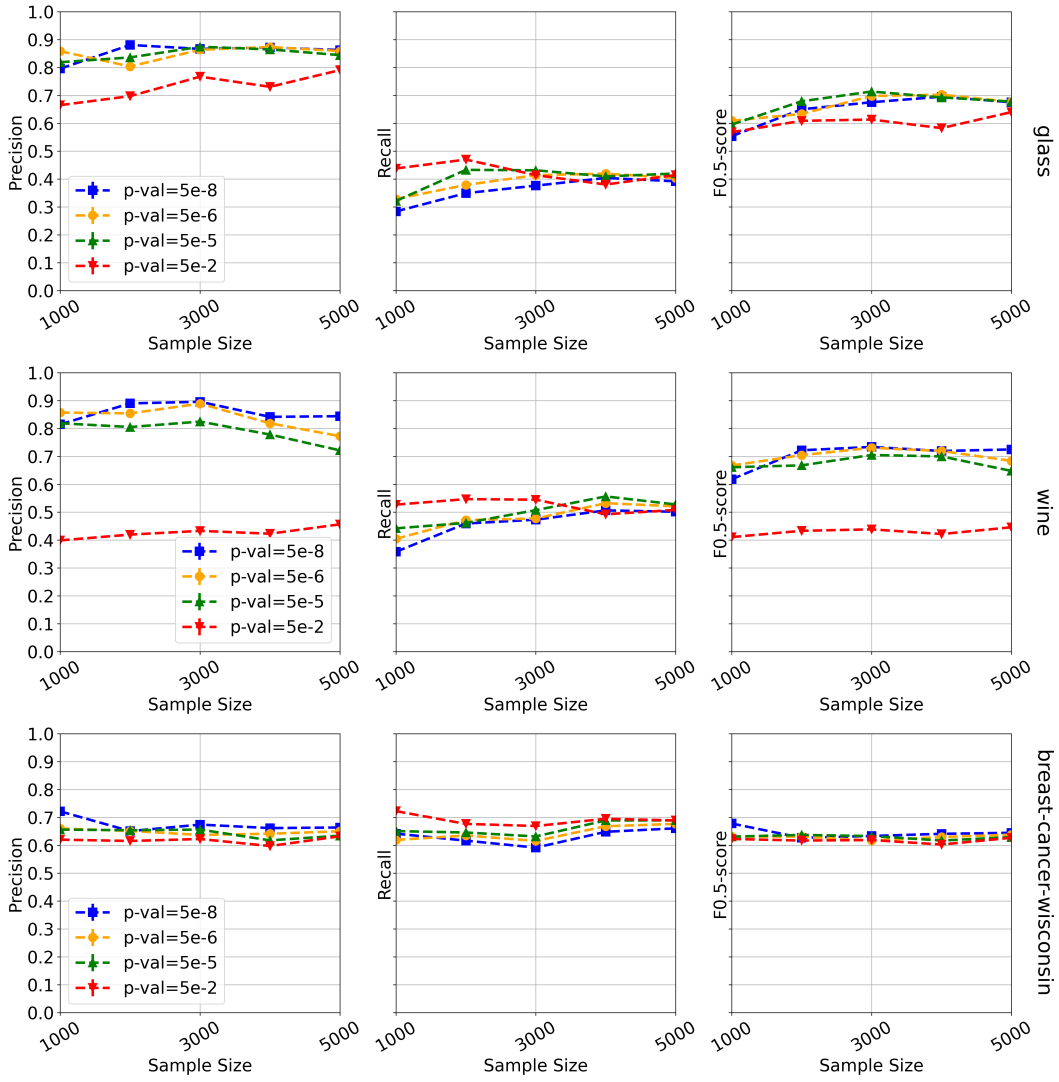

Figure 5.5: The effect of having a lower p-value threshold on the overall *Precision*, *Recall*, and $F_{0.5}$-*score*s. As it can be seen in the results, p-val $= 5e-8$ results in the best performance among the select datasets.

the tree generation algorithm at this point and begin the pruning step. This is in contrast with the original algorithm, which terminated when it added no new node after expanding a layer. We ran this modified version of the algorithm on the same datasets as in Section 4.4.1, and noticed this modification did not change the final rules the classifier created. This, however, did make the training time shorter. To figure out whether we could utilise the same approach in the explanation task, we conducted the same experiment and compared the explanations BARBE produced in both cases. Interestingly, we noticed a difference in the quality of the explanations, implying the rules that BARBE had created were also different. This indicates SigDirect, in some instances, creates additional rules in the tree even if there was no rule added in at least one of the earlier layers. We further evaluated these rules to see if this modification results in better explanations or worse. We present the results in Figure 5.6. The figures reveal that applying the early-stopping approach allows us to get a boost in *Precision*. We believe this is because the rules created in later layers in the tree are sometimes "overfitting" the classifier. Thus, an early-stopping approach like this can prevent it, eventually leading to more precise explanations.

5.6.3 Sampling Density

As we mentioned in the previous chapter, the goal of the sampling step in BARBE is to create a neighbourhood around the data point, so that BARBE can later use it to train an interpretable model. For this matter, we create a normal distribution around the bucket number of the original value for numerical features. We manage the density of the distribution by introducing a coefficient. We explore a few alternatives for this coefficient in this section. We report our findings in Figure 5.7. A larger coefficient means the distribution is denser, thus most synthetic data points in the sample belong to the proximity of the original data point. A smaller coefficient, however, indicates we observe data points that are farther away from the original data point more often.

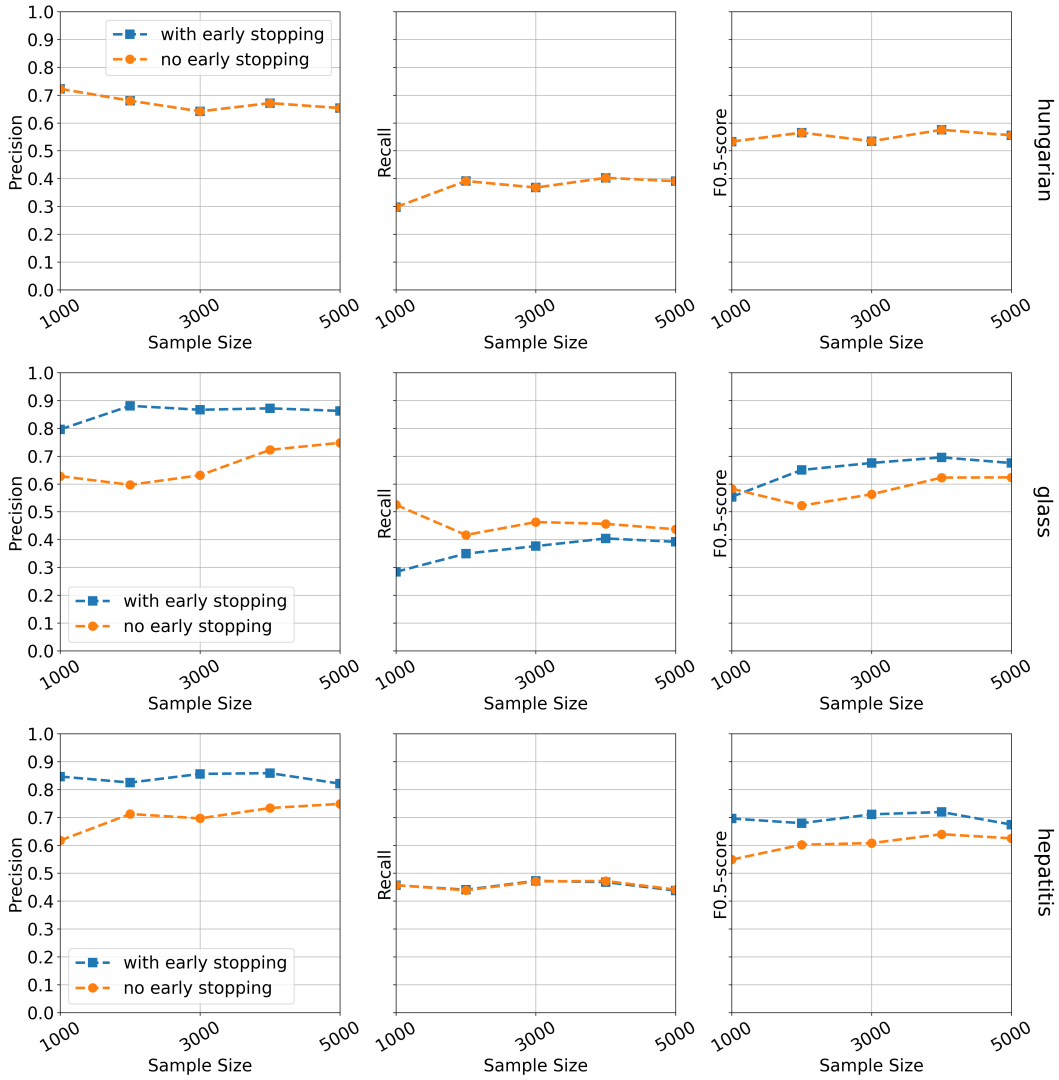

Figure 5.6: The effect of having the early-stopping on the overall *Precision*, *Recall*, and $F_{0.5}$-*score*s. As the results indicate, although the numbers are identical in some datasets, a few datasets get a lower *Precision* when there is no early-stopping.

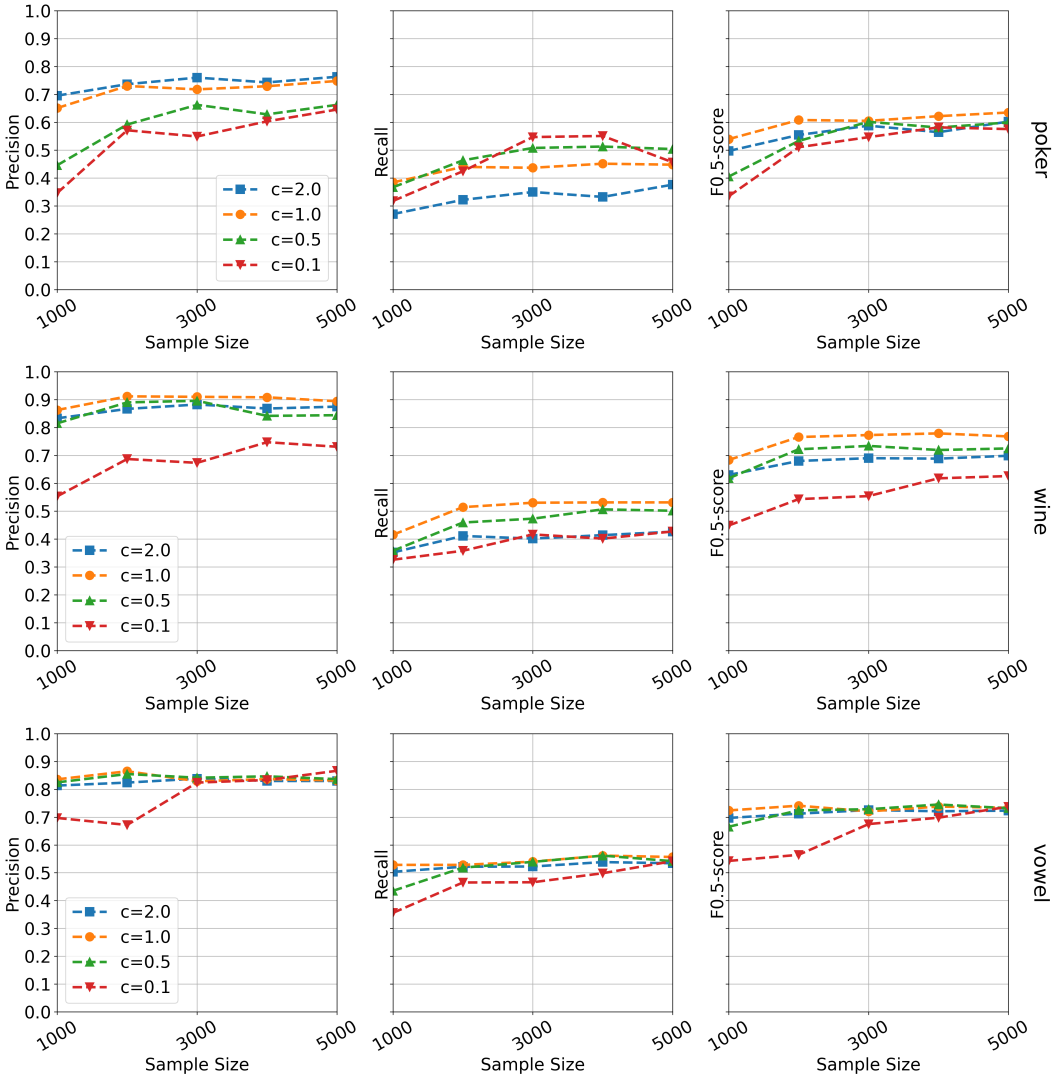

Figure 5.7: Various coefficients for sampling distribution can result in different explanation accuracies. When the coefficient is large (e.g., $c = 2.0$), BARBE can typically provide very precise explanations yet with low *Recall*. When the density is lower, on the other hand, *Recall* increases while *Precision* drops. As a trade-off, we decided to choose $c = 1.0$ as the default value in BARBE while the quality of explanations when $c = 0.5$ was also noteworthy.

5.6.4 Multi-class vs Binary Neighbourhoods

For multi-class datasets, since we are interested in explaining an instance that has a unique label, we can transform the dataset into a binary one where the new labels are 1) the label of the instance and 2) the others. This is called "One-vs-Rest" in the ML literature where many classifiers such as logistic regression have to transform multi-class datasets and build one such classifier for each class label to be able to handle multi-class datasets. While transforming a dataset into a binary one stems from a weakness in models such as logistic regression, SigDirect is capable of handling multi-class datasets natively. Despite this capability, we intend to know how having these two different datasets impacts the quality of the explanations produced by BARBE. We evaluate this idea on various datasets. We report the results in Figure 5.8. As the figures show, transforming the data into a binary dataset provides better results in terms of different metrics. Despite this advantage, we decided to keep multi-class datasets as they are. This is because it provides rules with proper class labels on their RHS. This is especially useful when we take advantage of them in BARBiE, the method we introduce in the next chapter.

5.6.5 Rule Selection

Once BARBE trains the interpretable model (SigDirect), it then selects the appropriate rules from the ruleset created by the model. In this step, there are multiple approaches to select the best rules. We can, for example, pick only the applied rules (i.e., applicable rules that their labels agree with the label of the instance). Alternatively, we can select the applied rules, followed by the rest of the applicable rules. Besides, we try to take advantage of the rest of the rules by selecting the ones that were very similar to the rest of the applicable rules yet had one feature that has a different value than the value in the original instance. This last one allows us to get a better *Recall* score compared to the first two alternatives. We provide the results for three different datasets In in Figure 5.9. Based on the figure, we can conclude that applied rules give us the best performance, especially in circumstances where

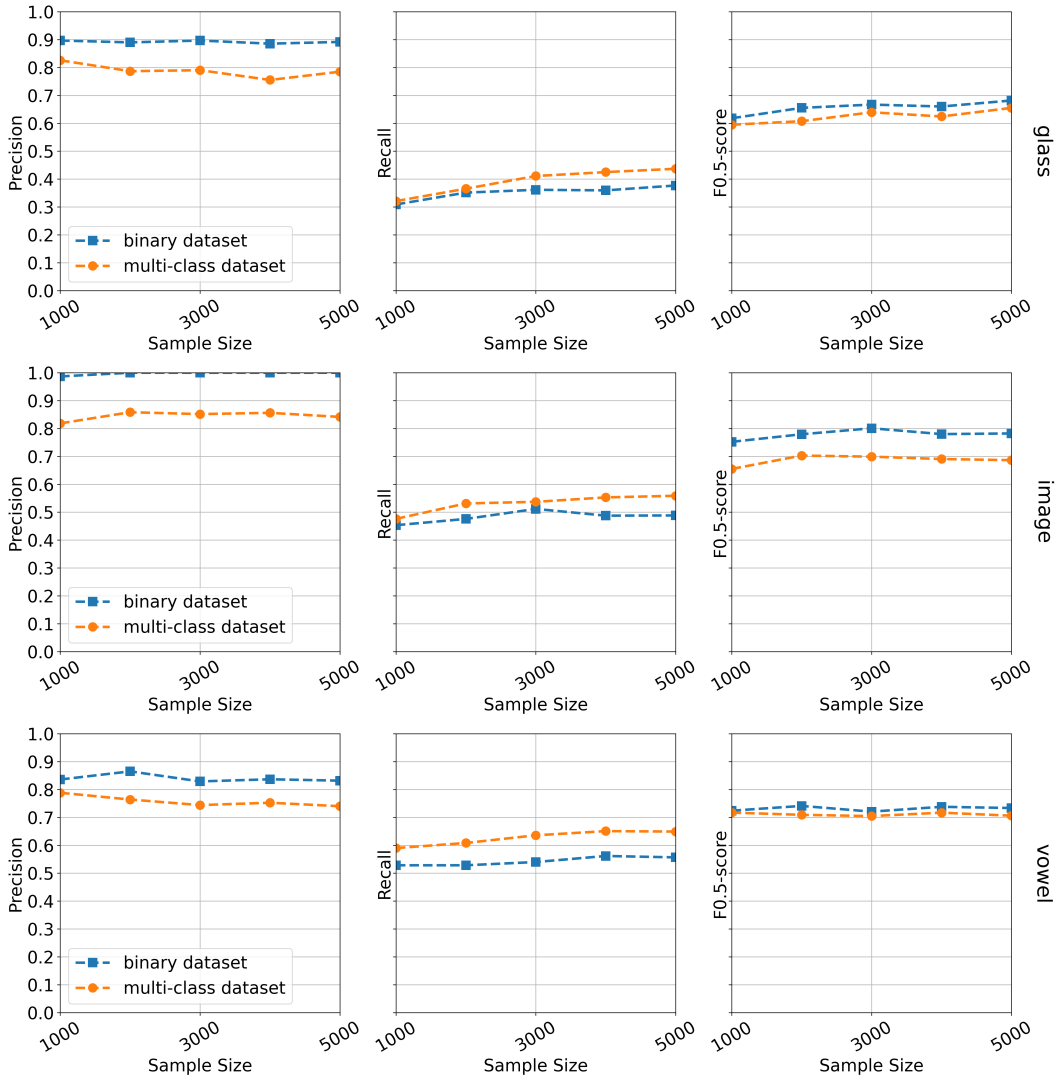

Figure 5.8: The effect of keeping multi-class datasets as they are against transforming them into a "One-vs-Rest" dataset. While changing the dataset to a binary dataset improves the *Precision*, it decreases the *Recall*.

Precision has higher importance than *Recall*.

5.6.6 Feature Extraction from Rules

As we discussed earlier, BARBE provides important features as another way of explaining the decisions of a black-box model. BARBE orders these features according to their importance. To better determine the quality of different strategies for feature ordering, we leverage the Rank-Biased Overlap (RBO) metric we discussed in Section 5.5.

This is in addition to the previous metrics we have taken advantage of so far. We report the results in Figure 5.10.

5.6.7 Improving Fidelity

We mentioned in Section 5.4.4 that if the class label predicted by any explanation framework does not agree with the black-box prediction, the explanation produced is not trustworthy. Based on this, BARBE does not produce any explanation unless the labels agree. In this section, we propose a simple way to improve fidelity by re-running BARBE should the labels disagree. This method simply re-samples from the original synthetic data points and ensures half of the selected data points belong to the target class while the rest do not. These data points are the ones that had been created and used in the previous try (i.e., *original* try as shown in Figure 5.11) and thus do not require the extra effort of labelling by the black-box model. Another method is to make a different distribution for data points and use that to train a SigDirect model. This alternative approach, however, requires labelling by the black-box model. We provide the results for our suggested method in Figure 5.11.

5.7 Comparison with Other Explainers

In this section, we compare our method against two competitive methods. As we mentioned in Chapter 3, LIME is the most-cited model-independent method that has been taken advantage of in different domains since its introduction, and many extensions to it have been introduced. Additionally,

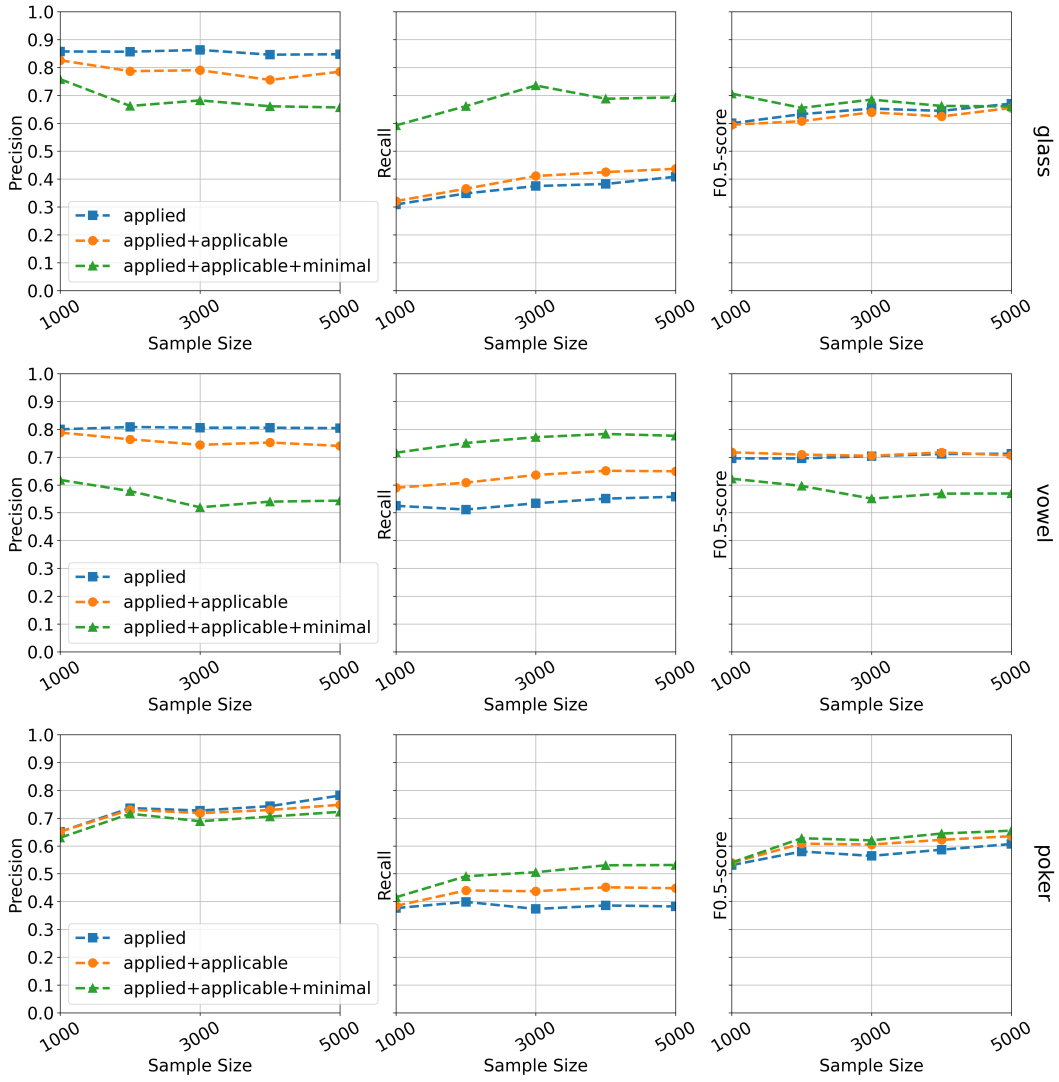

Figure 5.9: Different rule selection criteria result in different *Precision*, *Recall*, and $F_{0.5}$-*score*. Applied rules are a subset of applicable rules that the class labels agree with the class label of the instance. "Applied+Applicable" refers to the case where applied rules are added first and are followed by the rest of the applicable rules. Finally, the third reported results refer to the case where, in addition to the applied and the applicable rules, another set of rules are added. This new set of rules is very similar to applicable rules except that one feature on their LHS has a different value than that of the instance. These rules must have a different class label than the original instance.

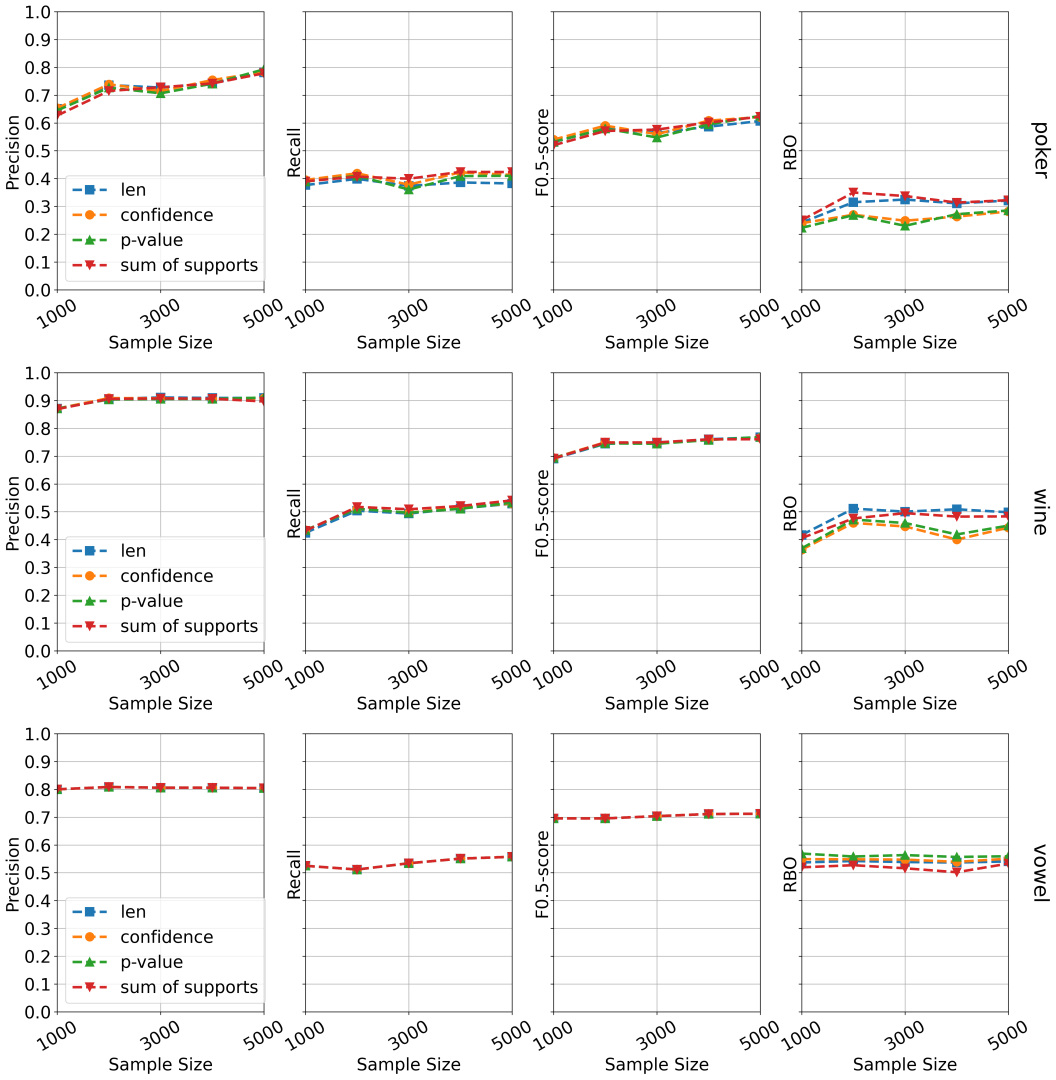

Figure 5.10: Different feature selection criteria result in different *Precision*, *Recall*, $F_{0.5}$, and *RBO* scores. "len" refers to the case where rules are sorted according to their number of LHS items (ascending), followed by multiplication of support and confidence scores. Finally, features are selected based on their first occurrence in the rules (the sooner they occur in the rules, the more important they are). The second and the third legends refer to a similar approach except that rules are sorted based on confidence scores (descending) and p-values (ascending), respectively. The fourth legend refers to the case where for each feature, we sum the support score of any rule they are present in their LHS. We then rank features according to these sums in descending order. Overall, the fourth approach provides better $F_{0.5}$, and *RBO* scores though the difference is not significant.

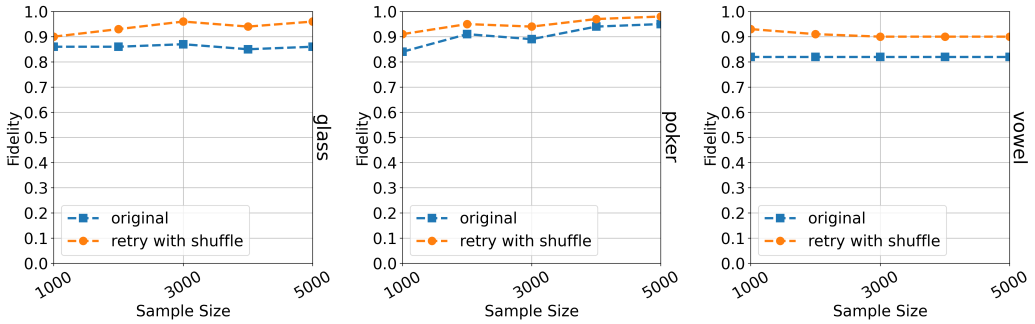

Figure 5.11: The fidelity scores for three different datasets are shown here. The first approach is shown in blue. The second one (i.e., orange marker) refers to the case where we reshuffle the synthetic data for instances incorrectly classified by SigDirect and make sure exactly half the data points belong to the target class. The figures show this approach benefits us in increasing the fidelity score, eventually helping BARBE provide explanations for more instances.

Anchor [60] is another method that benefits from rules to explain the decisions of a black-box model.

In Figure 5.12, we provide *Precision*, *Recall*, and $F_{0.5}$-*score* for all data points that the method has predicted the class label correctly. Besides, we also provide the RBO and fidelity scores separately. We exploit the test sets to conduct these experiments. We run each method with five different random seeds and report the averaged numbers.

Since LIME uses a regression model as its interpretable model, we examine its prediction score to verify the faithfulness of its model. If the score is below $1/n$, where n is the number of the classes, that explanation is not faithful, and thus we do not include that instance. Anchor, however, does not rely on any interpretable model, and therefore, we are not able to determine its fidelity. Consequently, we assume in our experiments that all its explanations are faithful, and as a result, we include all instances in the evaluation.

We also report the $F_{0.5}$ and *RBO* for all the datasets in Table 5.2 when the sample size is $5,000$[11].

[11]Please note that Anchor adaptively sets the number of samples required and thus this number does not apply to it.

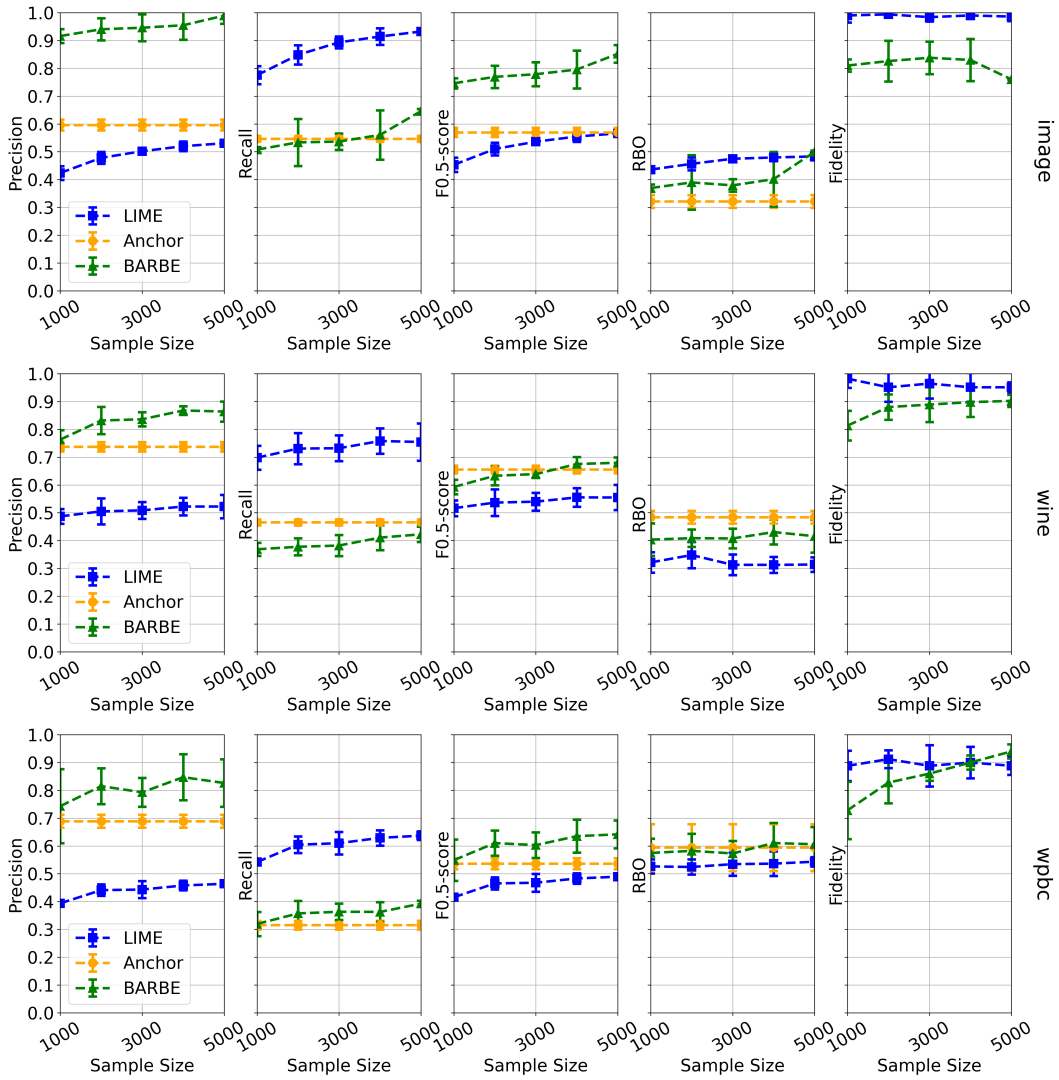

Figure 5.12: Performance of LIME, Anchor, and BARBE in different datasets.

	$F_{0.5}$-*score*			*RBO*		
dataset	**LIME**	**Anchor**	**BARBE**	**LIME**	**Anchor**	**BARBE**
Glass	0.736	0.534	**0.796**	0.777	0.287	**0.796**
Wine	0.554	0.656	**0.680**	0.314	**0.484**	0.416
Hungarian	0.483	0.508	**0.570**	**0.776**	0.264	0.427
Poker	**0.666**	0.331	0.637	0.353	**0.713**	0.368
Breast	0.633	0.497	**0.715**	0.300	0.246	**0.332**
Image	0.566	0.570	**0.852**	0.483	0.321	**0.500**
Magic	0.596	**0.729**	0.712	0.684	**0.835**	0.515
Vowel	0.526	0.683	**0.840**	0.671	0.444	**0.675**
Hepatitis	0.432	**0.492**	0.340	0.106	**0.218**	0.055
WPBC	0.489	0.536	**0.642**	0.543	0.594	**0.606**
WDBC	0.468	0.131	**0.494**	0.228	0.056	**0.320**

Table 5.2: $F_{0.5}$ and *RBO* for different methods across all datasets when the sample size is set to $5,000$.

5.8 Future Work

While BARBE can provide precise explanations as we showed, we believe it can produce even better explanations if we leverage more sophisticated sampling approaches. As we showed in Section 5.6.7, there are alternative approaches that can improve fidelity, and possibly the quality of explanations. We leave studying alternative sampling methods as future work.

Furthermore, Contrast Sets [63] is another interesting topic in Frequent Itemset Mining (where association rules come from), that we believe can benefit us in producing precise explanations. Contrast Sets learning takes advantage of association rules to find the most discriminative features between two classes in a dataset. If we apply this idea in the synthetic dataset we created in BARBE, we can discover the most distinctive features, which is essentially another way of producing explanations. The work of Jabbar et al. [34] is an interesting one where they also exploit statistical significance tests in discovering Contrast Sets in datasets. We can leverage that work to provide another way of explaining the decisions of black-box models.

Chapter 6

BARBiE: Black-box Association Rule-Based Interactive Explanations

As we showed in the previous chapter, BARBE can provide explanations for individual data points. The explanations provided by BARBE are in the form of association rules and important features. These two alternative types of explanations allow the end-user to better understand the rationale behind a system's decision. Here, we intend to introduce an interactive system, Black-box Association Rule-Based Interactive Explanations (BARBiE), which is an extension to BARBE. This extended framework allows the end-user to deepen their understanding of the decision-making process of the black-box model.

We only describe the capabilities of BARBiE and suggest some experiments for evaluation. We leave the implementation and experiments as future work.

We commence this chapter by reviewing the concept of interactive explanations in Section 6.1. Afterwards, we introduce BARBiE and review its advantages over other explanation frameworks, and discuss what capabilities it provides to end-users. Section 6.3 illustrates BARBiE with an example to show how it allows users to interact with the system. Lastly, we conclude this chapter in Section 6.4 where we recommend some experiments for consideration to evaluate the performance of BRABiE in a quantitative way.

6.1 Introduction

Atakishiyev et al. [4] propose different levels of explainability for XAI systems In their work, any system requires to have one or more attributes to belong to one level. One attribute of a Level Four XAI system, the ultimate level of explainability, is *interactivity*: the system needs to provide interactive explanations to the end-user.

The question that arises is what are the interactive explanations. Miller [45] discusses his viewpoint with an illustrative example where a system properly answers the questions raised by the user in each iteration. In his example, not only does the system base its answer on the question the user has asked, but it also takes into consideration the past conversation. The authors in [4] consider an explanation to be interactive if there are back-and-forth question and answers between the end-user and the system. The purpose of these dialogues is to build a deeper trust in the system. In other words, the goal is basically to let the end-user gain a better understanding of the system (i.e., how the system made the decision) by further revealing its decision-making process to them through interactions. In addition to building trust, in our view, the interaction can also be a means to adjust or fix a model. In other words, explainability is not only about providing explanations to make a model transparent but also to "debug" a model. In the case of BARBE, we provide rules with an explanation. The interaction can provide an opportunity to adjust those rules.

Numerous AI systems have benefited from interaction to achieve the goal of building trust. For example, authors of [56] introduced explainD in 2006. The goal of explainD was to provide different types of explanations such as feature importance. More significantly, explainD allowed end-users to modify values of a few features of the instance the system was explaining (i.e., change a feature like the "colour" from red to green) and ask for the impact it had on the prediction and the explanation. At this point, the system would provide them with the updated set of important features, and the new class label, should it have changed. Their method, nevertheless, could only explain a few

classifiers such as Naive Bayes, linear support vector machine (SVM), and Logistic Regression.

Moreover, Chodos and Zaiane introduced ARC-UI [11] to additionally let end-users modify values for attributes of rules a rule-based classifier used for inference, thus enabling end-users to inject their domain expertise into the system.

While the two systems mentioned above worked to some extent on interpretable systems, the focus of recent work in XAI, however, is on explaining black-box models.

6.2 BARBiE

BARBiE allows the user to not only change the values of important features but also change the values for non-important ones. This is what many call "what-if" analysis. Moreover, BARBiE allows them to find out about the features in which modifying them results in different class labels. Finally, it provides the user to have an "editable" classifier where through interaction, the user should be able to inject domain knowledge into the model. We discuss these capabilities in more detail in this section.

6.2.1 What-if analysis

As we mentioned earlier, one way to provide interactive explanations is by allowing end-users to assess the implications of modifying feature values for each data point. This means the explanation system should be able to provide explanations for any new data point that resulted from a modification the end-user made to the original data point.

Although rules are preferred over feature importance scores employed by methods such as LIME, as we discussed earlier, that is not the only advantage an association rule-based classifier has over sparse linear models. In a sparse linear model such as linear regression, coefficients in the model are the only pieces forming the classifier. Therefore, they are the only source of information the explanation framework can profit from when answering questions of end-

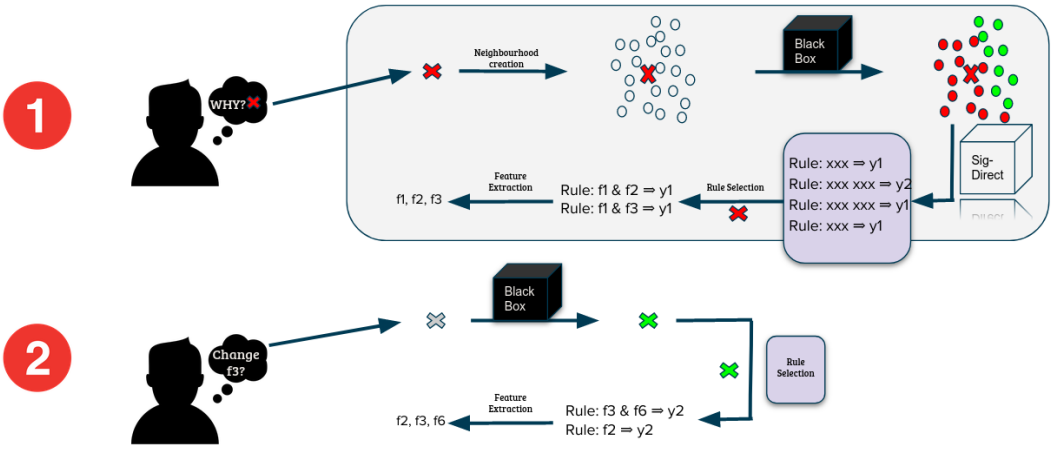

Figure 6.1: Different steps in BARBiE. Note that the purple box in step two is the same as the purple box in step one, and both contain the ruleset created by SigDirect.

users. As a result, LIME needs to produce a new explanation for nearby data points as if they were independent data points, decreasing the usefulness of interaction for users[1]. For a rule-based classifier such as SigDirect, however, only a few of the rules (i.e., applied rules) it has created are what it utilises to explain the decision of the black-box model. Hence, we can take advantage of the whole ruleset and produce explanations for other data points (i.e., find the applied rules for each data point and create the corresponding explanation).

Moreover, the synthetic dataset BARBE had generated was around the original instance the user had requested an explanation for. Thus, there shall be isolated data points that are far away from it. This indicates there may not be any applied rule available for distant data points, and if any, they may not be reliable rules especially given their low support scores. Nevertheless, if BARBiE determines its explanation for that data point is not reliable, it can resample and provide the appropriate explanation ultimately (i.e., run BARBE over the new data point).

In other words, if the end-user is curious to comprehend what happens if they alter one or more features in a data point the system has already explained, it can instantly provide the proper explanation without doing extra

[1]In our view, this would not count as an interactive explanation anymore.

computations such as labelling a new synthetic dataset[2]. This is because the system already has the required rules to explain the new data point provided it is not far from the original one. Figure 6.1 illustrates these two different situations. In the first step, BARBiE calls BARBE and provides the appropriate explanation. Afterwards, it exploits the same trained SigDirect model to provide explanations for data points that are in the vicinity of the original point to be explained. If the user requests explanation for data points that are far, however, it would have to call BARBE again to sample another subspace (step one).

6.2.2 Counter-factual examples

One more interesting question some meticulous users would ask a system is regarding data points similar to the original instance yet with different class labels. These are, basically, data points that are very similar to the original one (most features have the same values) but the class labels are different. Reviewing these data points would allow end-users to know what changes they need to make to the data point to alter the class label. Moreover, it helps them get a better insight into the black-box model. Researchers use the term "counter-factual" for such examples.

As we mentioned in Section 5.3.2 BARBE creates synthetic data points in the vicinity of the instance. It then "undiscretizes" them to subsequently label them using the black-box model. The "undiscretized" data points that have different labels than the original data point are counter-factual examples. BABRiE calculates a distance score with respect to the original instance for each data point and then sorts them according to the score. Consequently, the end-user can observe the counter-factual examples in an ordered list where the first instance is the closest one to the original data point. Besides, BARBiE generates the appropriate explanation for that data point (according to the way we mentioned in the previous section) as an additional piece of information for the user, further helping them understand the new data point. The explanation for such counter-factual examples, containing a set of rules, helps the

[2]This may not be possible in some circumstances.

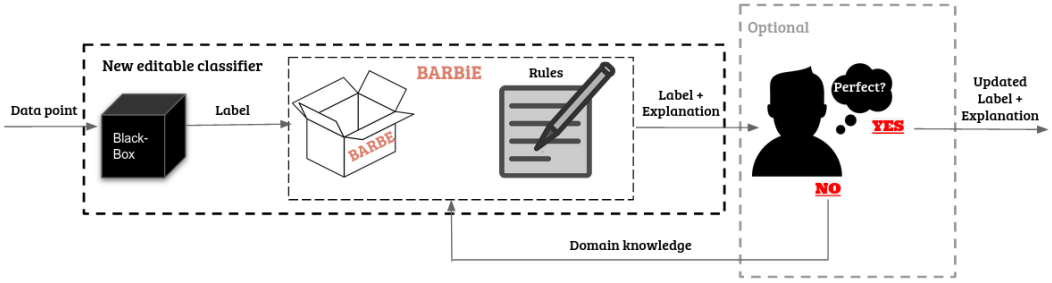

Figure 6.2: A proposed architecture for editable classifiers. In this new classifier, any black-box model could be used under-the-hood.

user not only understand which features were significant in the modification of the class label but also what associations exist among the features.

6.2.3 Editable Classifiers

With BARBE, we used rules to explain the decisions of a black-box model. Additionally, we guaranteed the fidelity of our system by enforcing it to have the same class label as the black-box model. We can leverage these two facts and have a new *editable classifier* which is based on the current black-box model. Unlike "readable" classifiers that we cannot change once they are trained, we can modify the inference behaviour of editable classifiers directly any time we desire. Note that this is different than online learning where the user can update the model by using new data points. In editable models, however, the user injects their knowledge into the system directly, without requiring new data points.

The new classifier illustrated in Figure 6.2 contains a pipeline where once the black-box model has completed its inference, BARBiE is executed to get the corresponding explanation. At this point, the rules in the explanation are compared with the domain knowledge that exists. If the domain knowledge contradicts any of those rules, a modified version of rules (which are based on the domain knowledge) are executed in SigDirect to obtain the new class label. Note that since the SigDirect model in BARBE was faithful to the initial model, as we had guaranteed so, the new label will also be faithful. The domain knowledge is in the form of rules which are created by a domain

expert. Additionally, the user can analyze the output of the classifier (which includes an explanation) and if needed, define new rules to modify the system's behaviour, provided they are an expert in the domain. Consequently, the classifier is capable of detecting complex patterns in the data (as it incorporates a black-box model), yet domain knowledge could also be injected into it by a domain expert should the expect want.

6.3 Example of Interaction with BARBiE

To further demonstrate how BARBiE can help users by interactively explaining the decisions of a black-box model, we provide an example in this section. The example, illustrated in Figure 6.3, is about a toy dataset where we want to classify used cars as cheap or expensive according to some of their features such as year they are built, colour, make, and model. In the beginning, there is a black 2010 Toyota Corolla car classified by the black-box model as cheap. The user is interested to know why the model has predicted this car as cheap. As a result, they ask for an explanation from BARBiE. The explanation framework explains this instance by providing the relevant explanation. In the second and the third iterations, the user is interested to know how the class labels and the explanations would differ if the colour or the model and the make of the car were different. Then, BARBiE provides the appropriate explanations for these two instances without needing to run BARBE under-the-hood as it is sufficient to extract the relevant rules from the ruleset created earlier. The last question the user may ask is regarding a very similar car to the black 2010 Toyota Corolla but with a different class label. BARBiE looks for cars with minimum modifications yet with a different label. The explainer achieves this by searching through the synthetic instances labelled by the black-box model it had created for explaining the original car. It, thus, does not need to query the black-box model again in this step either.

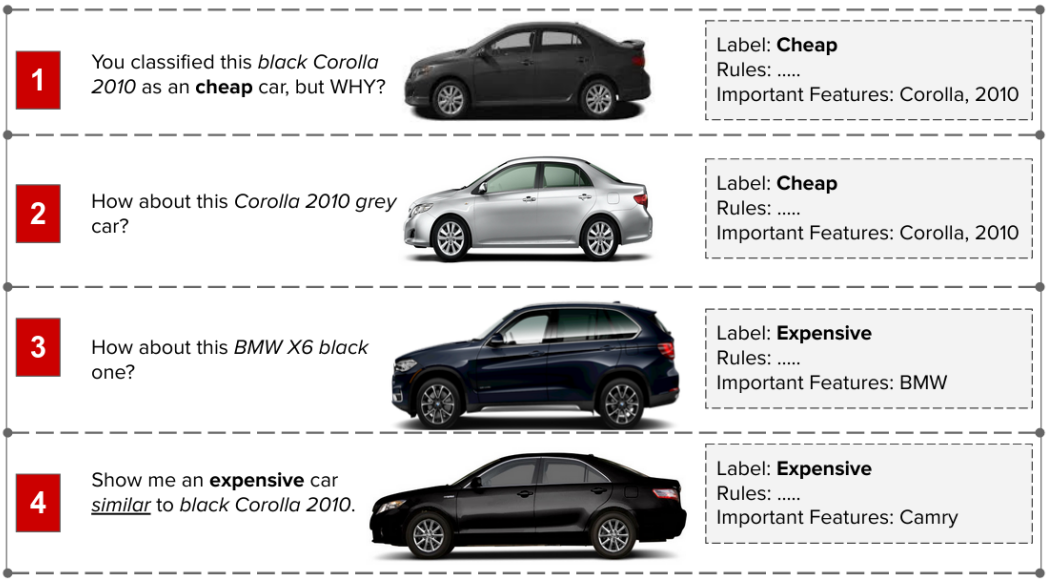

Figure 6.3: This figure presents an example of how BARBiE interacts with end-users. In this toy example, the user asks why the black-box has classified the black 2010 Toyota Corolla as cheap. BARBiE provides the corresponding explanation in the form of rules and important features. Then, the user modifies the colour feature of the instance and then requests prediction and its explanation. In the third step, the user changes the model and make of the car. Afterwards, BARBiE provides the appropriate explanation, together with the class label obtained from the black-box model. Finally, in case four the user requests a similar car to the one it had queried initially in which is not cheap anymore (i.e., counter-factual example). At this point, BARBiE suggests a black 2010 Toyota Camry, which is classified expensive by the black-box model, yet is similar to the one the user was requesting.

6.4 Wrap up

As we discussed in the previous section, since the initial sample is created around the original data point, BARBiE can explain the nearby data points without requiring to run BARBE again. In other words, we can find the appropriate applied rules in the ruleset for data points up to a known distance while explanations for points beyond that distance are not accurate. Furthermore, applied rules for data points that are even closer than the distant ones may have low support scores. BARBiE cannot explain these data points properly with the trained SigDirect model. Determining the maximum distance BARBiE can provide reliable explanations can be the objective of an experiment. In this experiment, we begin from the original data point and gradually increase the number of modified features (compared to the initial values), and compare the produced explanations against the ground truth data, and observe how the quality of the experiments deteriorates.

Another set of experiments can depend on humans to perceive how BARBiE benefits them to deepen their trust in the predictions of the black-box model. To evaluate the helpfulness of BARBiE, one can, for example, measure the satisfaction of users when provided with BARBiE explanations, compared to random explanations. Another experiment can examine how providing explanations for an accurate black-box model can help users distinguish it from a random classifier.

Chapter 7

Conclusion

The field of eXplainable Artificial Intelligence (XAI) has attracted huge interest in recent years. Many researchers have concentrated on introducing different explanation frameworks that provide explanations for different Deep Neural Network (DNN) architectures. Although these methods may be capable of performing reasonably well in terms of accuracy, they all are limited to specific architectures in DNNs. Ribeiro et al. [59] introduced LIME as a framework that provides explanations for any black-box model. While they claimed LIME was completely model-agnostic (i.e., users could apply it on any model), their method requires the black-box model to provide a probability score for each class. Additionally, the quality of the explanations LIME provides and how it performs across different datasets remained an open problem. Furthermore, they provided explanations in the form of feature importance scores, where the scores were not necessarily useful to the users.

In this work, we reviewed LIME in Chapter 3 in details, and then conducted a few experiments on it. We demonstrated that the quality of the explanations produced by LIME varies across different datasets. Additionally, we showed how selecting different discretization strategies in LIME results in different outcomes with no single winner.

In Chapter 5, we introduced Black-box Association Rule-Based Explanation (BARBE), the main contribution of this work that works on tabular datasets. BARBE is a model-agnostic explanation framework that provides highly precise explanations in the form of rules, in addition to an ordered set of

important features. These rules can help users not only identify features that played a critical role in the decision-making process of the black-box but also notice the associations among these features. Unlike LIME, however, BARBE can handle any black-box classifier and does not depend on the probability scores produced by some classifiers. In contrast with LIME, we defined a faithfulness measure for BARBE and directly verified its fidelity concerning the black-box model. We used *Precision*, *Recall*, and $F_{0.5}$-*score* to evaluate the explanations produced by BARBE on multiple datasets. We showed BARBE beats both LIME and Anchor [60] in most datasets in terms of $F_{0.5}$-*score*. Interestingly, BARBE outperforms Anchor in not only $F_{0.5}$-*score*, but also *Precision* in most datasets, even though Anchor depends on a high precision rule to explain an instance.

We took advantage of Ranked-Biased Overlap (RBO) [69] to compare the order in which features were ranked in different explanation frameworks. We demonstrated through many experiments that BARBE bests both LIME and Anchor in this metric as well.

Moreover, we introduced Black-box Association Rule-Based Interactive Explanations (BARBiE) in Chapter 6 as an extension to BARBE. BARBiE allows users to receive interactive explanations in the form of "what-if" analysis, in addition to counter-factual examples. These interactive explanations allow the user to ask BARBiE what happens if they modify the value of one or more features. BARBiE would respond by showing how such changes would affect the class label prediction and the explanation. These two allow the user to grasp the underlying decision-making process of the classifier, helping them gain a deeper trust in the system.

As we mentioned in Section 5.8, exploring the concept of Contrast Sets and applying it to black-box models to extract explanations is an interesting approach that we leave as future work. In addition to that, constructing the appropriate sample, in our view, plays a vital role in producing the right explanation. Although we explored one way to construct the neighbourhood in this work, other strategies such as the Genetic Algorithm may lead to improved results. Finally, modifying BARBE and BARBiE to make it capable of

processing other data types (e.g., text, images) is another direction that can extend this work.

References

[1] R. Agrawal, R. Srikant, *et al.*, "Fast algorithms for mining association rules," in *Proc. 20th int. conf. very large data bases, VLDB*, vol. 1215, 1994, pp. 487–499. 46

[2] *Amazon doesn't consider the race of its customers. should it?* [Online]. Available: `https://www.bloomberg.com/graphics/2016-amazon-same-day/`. 10

[3] M.-L. Antonie and O. R. Zaiane, "Text document categorization by term association," in *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, IEEE, 2002, pp. 19–26. 48

[4] S. Atakishiyev, H. Babiker, N. Farruque, R. Goebel, M. Kima, M. Motallebi, J. Rabelo, T. Syed, and O. Zaiane, "A multi-component framework for the analysis and design of explainable artificial intelligence," *arXiv preprint arXiv:2005.01908*, 2020. iv, 13, 14, 20, 29, 83

[5] H. K. B. Babiker and R. Goebel, "Using kl-divergence to focus deep visual explanation," *arXiv preprint arXiv:1711.06431*, 2017. 14, 29

[6] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014. 22, 23

[7] O. Biran and C. Cotton, "Explanation and justification in machine learning: A survey," in *IJCAI-17 workshop on explainable AI (XAI)*, vol. 8, 2017. 11

[8] R. P. Brazile and K. M. Swigger, "Gates: An airline gate assignment and tracking expert system," *IEEE Intelligent Systems*, no. 2, pp. 33–39, 1988. 2

[9] B. G. Buchanan, "Expert systems: Working systems and the research literature," *Expert systems*, vol. 3, no. 1, pp. 32–50, 1986. 2

[10] R. Challen, J. Denny, M. Pitt, L. Gompels, T. Edwards, and K. Tsaneva-Atanasova, "Artificial intelligence, bias and clinical safety," *BMJ Qual Saf*, vol. 28, no. 3, pp. 231–237, 2019. 9

[11] D. Chodos and O. R. Zaiane, "Arc-ui: Visualization tool for associative classifiers," in *2008 12th International Conference Information Visualisation*, IEEE, 2008, pp. 296–301. 84

[12] E. Choi, M. T. Bahadori, J. Sun, J. Kulas, A. Schuetz, and W. Stewart, "Retain: An interpretable predictive model for healthcare using reverse time attention mechanism," in *Advances in Neural Information Processing Systems*, 2016, pp. 3504–3512. 23

[13] W. W. Cohen, "Fast effective rule induction," in *Machine learning proceedings 1995*, Elsevier, 1995, pp. 115–123. 46

[14] M. Craven and J. W. Shavlik, "Extracting tree-structured representations of trained networks," in *Advances in neural information processing systems*, 1996, pp. 24–30. 21

[15] J. Dastin, *Amazon scraps secret ai recruiting tool that showed bias against women*, Oct. 2018. [Online]. Available: `https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G`. 10

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255. 27

[17] J. Doe, *Darpa xai baa*, `https://www.darpa.mil/attachments/DARPA-BAA-16-53.pdf`. (visited on 04/20/2019). 11, 14, 15

[18] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017. 12, 27

[19] F. K. Došilović, M. Brčić, and N. Hlupić, "Explainable artificial intelligence: A survey," in *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*, IEEE, 2018, pp. 0210–0215. 11

[20] D. Dua and C. Graff, *UCI machine learning repository*, 2017. [Online]. Available: `http://archive.ics.uci.edu/ml`. 32, 41, 52, 63

[21] *Equal credit opportunity act*, Jan. 2020. [Online]. Available: `https://en.wikipedia.org/wiki/Equal_Credit_Opportunity_Act`. 11

[22] T. Fawcett and F. Provost, "Adaptive fraud detection," *Data mining and knowledge discovery*, vol. 1, no. 3, pp. 291–316, 1997. 3

[23] E. A. Feigenbaum, B. G. Buchanan, and J. Lederberg, "On generality and problem solving: A case study using the dendral program," 1970. 2

[24] *General data protection regulation*, Feb. 2020. [Online]. Available: `https://en.wikipedia.org/wiki/General_Data_Protection_Regulation`. 4, 10

[25] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti, "Local rule-based explanations of black box decision systems," *arXiv preprint arXiv:1805.10820*, 2018. 29, 38, 39, 56

[26] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM computing surveys (CSUR)*, vol. 51, no. 5, p. 93, 2018. 16, 21

[27] P. Hall, N. Gill, M. Kurka, and W. Phan, *Machine learning interpretability with h2o driverless ai*, 2019. 38

[28] W. Hamalainen, "Efficient discovery of the top-k optimal dependency rules with fisher's exact test of significance," in *2010 IEEE International Conference on Data Mining*, IEEE, 2010, pp. 196–205. 50

[29] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *ACM sigmod record*, ACM, vol. 29, 2000, pp. 1–12. 46

[30] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell, "Generating visual explanations," in *European Conference on Computer Vision*, Springer, 2016, pp. 3–19. 4

[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 5

[32] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Machine learning*, vol. 11, no. 1, pp. 63–90, 1993. 45

[33] L. Hu, J. Chen, V. N. Nair, and A. Sudjianto, "Locally interpretable models and effects based on supervised partitioning (lime-sup)," *arXiv preprint arXiv:1806.00663*, 2018. 38, 39

[34] M. S. M. Jabbar and O. R. Zaiane, "Learning statistically significant contrast sets," in *Canadian Conference on Artificial Intelligence*, Springer, 2016, pp. 237–242. 81

[35] S. Jain and B. C. Wallace, "Attention is not explanation," *arXiv preprint arXiv:1902.10186*, 2019. 23

[36] H. Lakkaraju, E. Kamar, R. Caruana, and J. Leskovec, "Interpretable & explorable approximations of black box models," *arXiv preprint arXiv:1707.01154*, 2017. 15

[37] J. Li and O. R. Zaiane, "Exploiting statistically significant dependent rules for associative classification," *Intelligent Data Analysis*, vol. 21, no. 5, pp. 1155–1172, 2017. 49, 51–53, 69

[38] W. Li, J. Han, and J. Pei, "Cmar: Accurate and efficient classification based on multiple class-association rules," in *Proceedings 2001 IEEE international conference on data mining*, IEEE, 2001, pp. 369–376. 48

[39] S.-H. Liao, "Expert system methodologies and applications—a decade review from 1995 to 2004," *Expert systems with applications*, vol. 28, no. 1, pp. 93–103, 2005. 1

[40] Z. C. Lipton, "The mythos of model interpretability," *arXiv preprint arXiv:1606.03490*, 2016. 12

[41] B. Liu, W. Hsu, Y. Ma, *et al.*, "Integrating classification and association rule mining.," in *KDD*, vol. 98, 1998, pp. 80–86. 47

[42] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*, 2017, pp. 4765–4774. 15, 24, 29, 33

[43] J. McDermott, "R1: The formative years," *AI magazine*, vol. 2, no. 2, p. 21, 1981. 1

[44] E. Mendelson, *Introduction to Mathematical Logic, 6th Edition*. CRC Press, 2015. 13

[45] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial Intelligence*, 2018. 11, 26, 56, 83

[46] S. Mishra, B. L. Sturm, and S. Dixon, "Local interpretable model-agnostic explanations for music content analysis.," in *ISMIR*, 2017, pp. 537–543. 30

[47] C. Modarres, M. Ibrahim, M. Louie, and J. Paisley, "Towards explainable deep learning for credit lending: A case study," *arXiv preprint arXiv:1811.06471*, 2018. 30

[48] C. Molnar *et al.*, "Interpretable machine learning: A guide for making black box models explainable," *Christoph Molnar, Leanpub*, 2018. 16, 36

[49] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1–15, 2018. 12

[50] A. Moore, Y. Cai, K. Jones, and V. Murdock, "Tree ensemble explainability," 2017. 21

[51] M. Moravčík, M. Schmid, N. Burch, V. Lisý, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. Bowling, "Deepstack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, no. 6337, pp. 508–513, 2017. 3

[52] L. Munkhdalai, L. Wang, H. W. Park, and K. H. Ryu, "Advanced neural network approach, its explanation with lime for credit scoring application," in *Asian Conference on Intelligent Information and Database Systems*, Springer, 2019, pp. 407–419. 30

[53] J. A. Nelder and R. W. Wedderburn, "Generalized linear models," *Journal of the Royal Statistical Society: Series A (General)*, vol. 135, no. 3, pp. 370–384, 1972. 16

[54] H. Núñez, C. Angulo, and A. Català, "Rule extraction from support vector machines.," in *Esann*, 2002, pp. 107–112. 21

[55] 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. 64

[56] B. Poulin, R. Eisner, D. Szafron, P. Lu, R. Greiner, D. S. Wishart, A. Fyshe, B. Pearcy, C. MacDonell, and J. Anvik, "Visual explanation of evidence with additive classifiers," in *Proceedings of the National Conference on Artificial Intelligence*, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, vol. 21, 2006, p. 1822. 83

[57] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014. 17

[58] ——, "Learning logical definitions from relations," *Machine learning*, vol. 5, no. 3, pp. 239–266, 1990. 45

[59] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, ACM, 2016, pp. 1135–1144. 5, 15, 24, 29, 91

[60] ——, "Anchors: High-precision model-agnostic explanations," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 25, 26, 29, 56, 79, 92

[61] C. Rudin, "Please stop explaining black box models for high stakes decisions," *arXiv preprint arXiv:1811.10154*, 2018. 15

[62] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited, 2016. 1

[63] A. Satsangi and O. R. Zaiane, "Contrasting the contrast sets: An alternative approach," in *11th International Database Engineering and Applications Symposium (IDEAS 2007)*, IEEE, 2007, pp. 114–119. 81

[64] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 618–626. 4, 23, 27, 29

[65] L. S. Shapley, "A value for n-person games," *Contributions to the Theory of Games*, vol. 2, no. 28, pp. 307–317, 1953. 24

[66] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017. 3

[67] K. Sukel, *Artificial intelligence ushers in the era of superhuman doctors*, Jul. 2017. [Online]. Available: `https://www.newscientist.com/article/mg23531340-800-artificial-intelligence-ushers-in-the-era-of-superhuman-doctors/`. 10

[68] S. H. Villarroya, "Interpretability in sequence tagging models for named entity recognition," 2018. 31

[69] W. Webber, A. Moffat, and J. Zobel, "A similarity measure for indefinite rankings," *ACM Transactions on Information Systems (TOIS)*, vol. 28, no. 4, pp. 1–38, 2010. 67, 92

[70] R. Wexler, "When a computer program keeps you in jail: How computers are harming criminal justice," *New York Times*, 2017. 3

[71] L. S. Whitmore, A. George, and C. M. Hudson, "Mapping chemical performance on molecular structures using locally interpretable explanations," *arXiv preprint arXiv:1611.07443*, 2016. 30

[72] M. Wilcox, S. Schuermans, and C. Voskoglou, "Developer economics, state of the developer nation," Q3 2016. Technical report, VisionMobile Ltd, Tech. Rep., 2016. 52

[73] X. Yin and J. Han, "Cpar: Classification based on predictive association rules," in *Proceedings of the 2003 SIAM International Conference on Data Mining*, SIAM, 2003, pp. 331–335. 48

Appendix A

Other Experiments Figures

In this appendix, we provide the results of experiments we conducted to tune BARBE for all datasets in the next sections. Each section contains the results for one experiment. The last section contains the results for the experiments conducted to compare BARBE against LIME and Anchor, the two competing approaches.

A.1 P-value Threshold in SigDirect

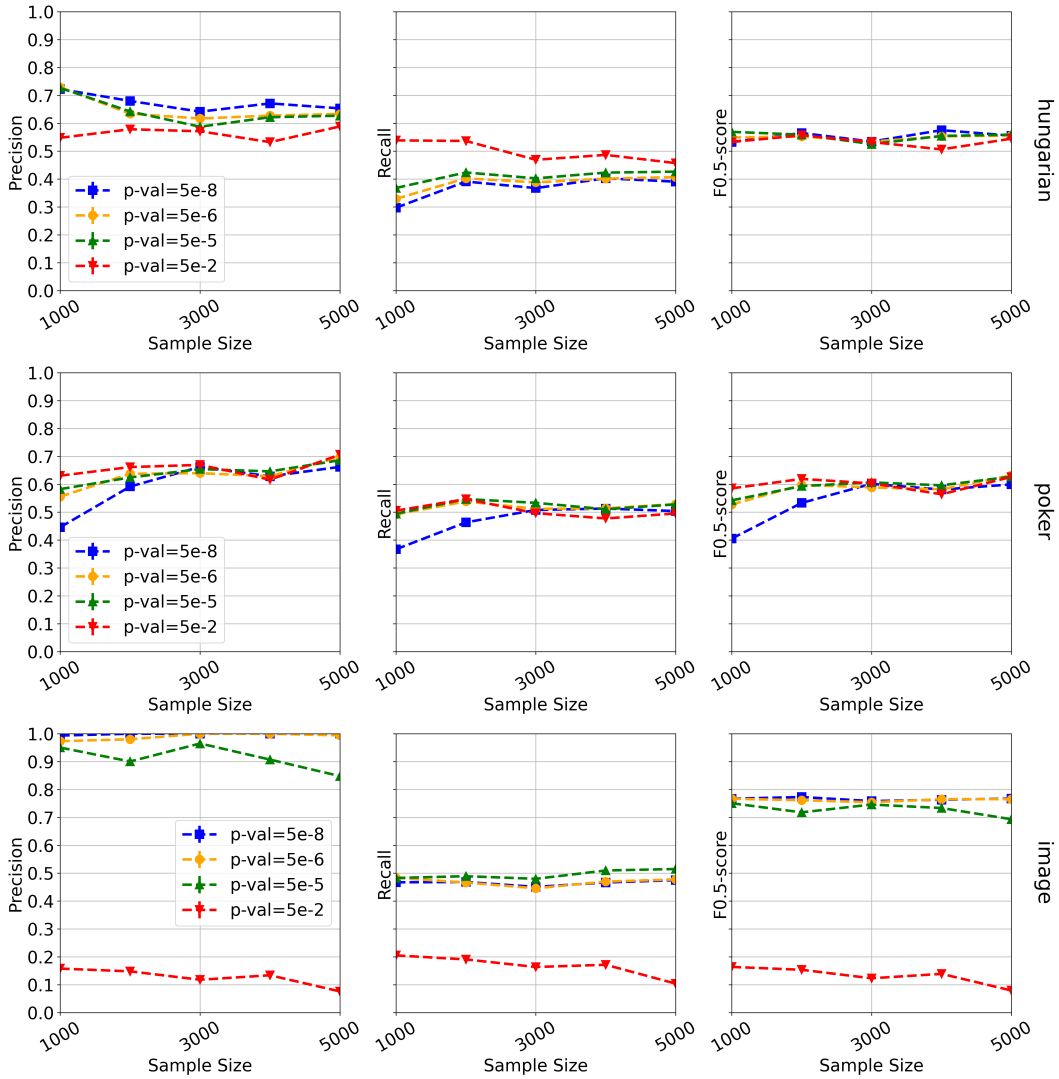

Figure A.1: The effect of having a lower p-value threshold on the overall *Precision*, *Recall*, and $F_{0.5}$-*score*s.

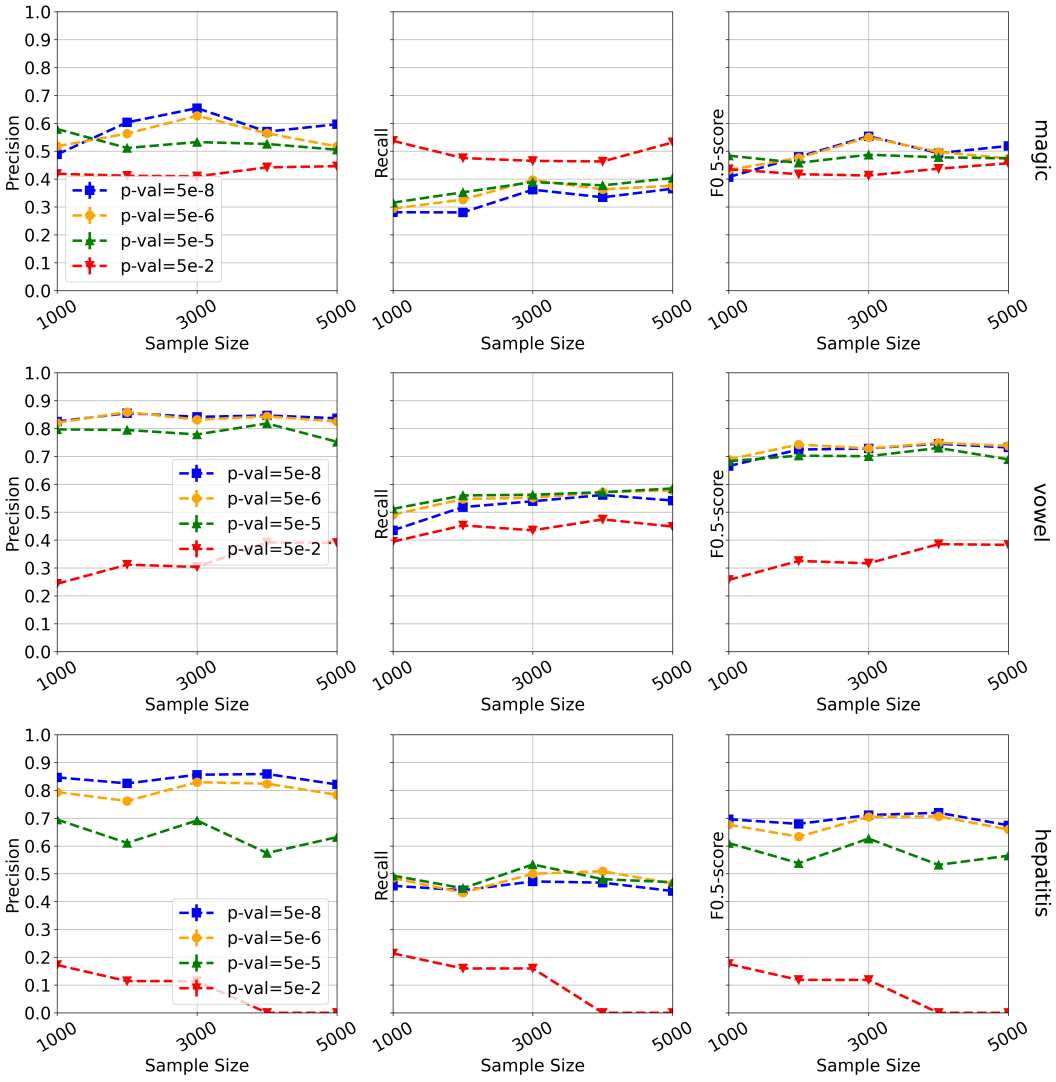

Figure A.1: The effect of having a lower p-value threshold on the overall *Precision*, *Recall*, and $F_{0.5}$-*score*s.

A.2 Early-Stopping in SigDirect

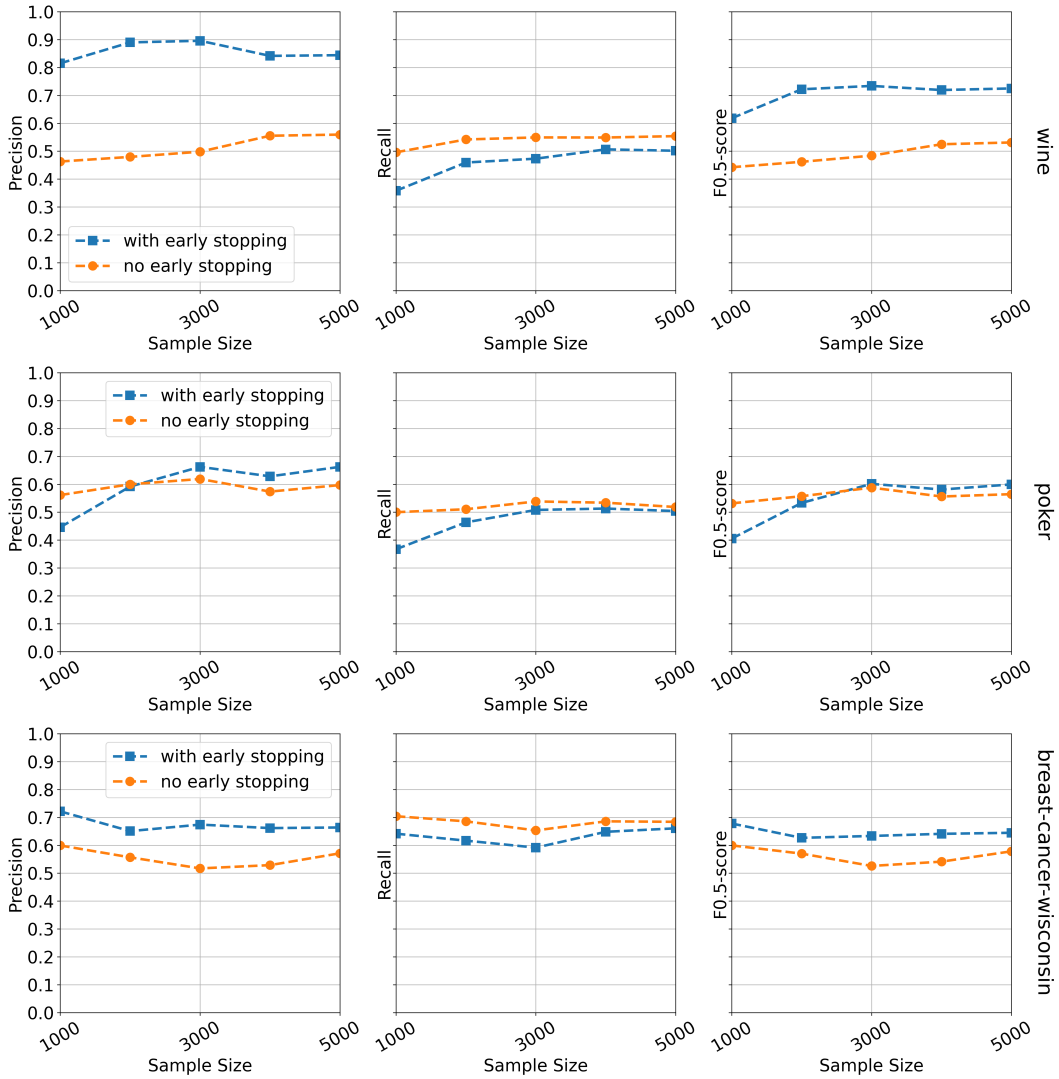

Figure A.2: The effect of having the early-stopping on the overall *Precision*, *Recall*, and $F_{0.5}$-*score*s. Note that we did not continue the experiment for pval=5e-2 on Hepatitis dataset as it was slow and results were not promising.

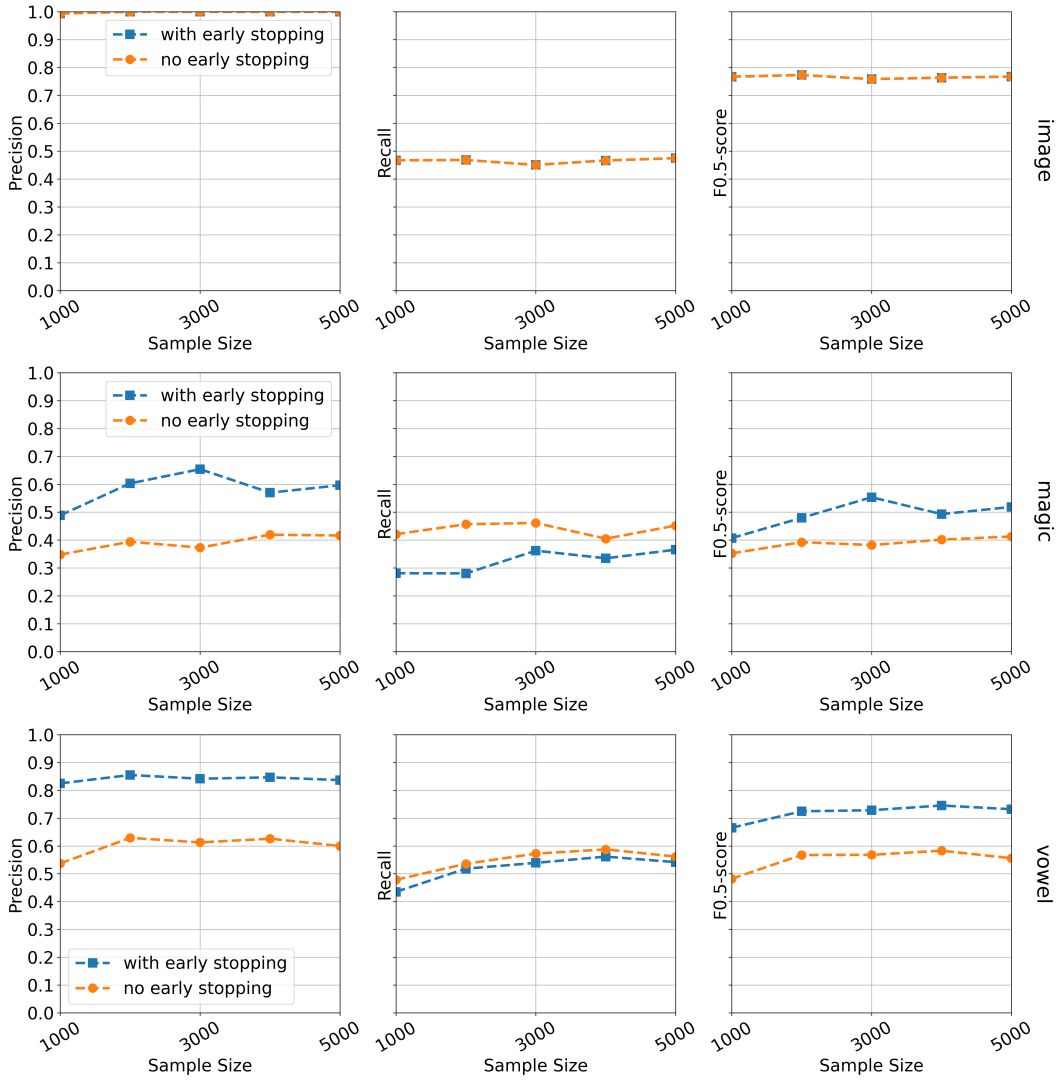

Figure A.2: The effect of having the early-stopping on the overall *Precision*, *Recall*, and $F_{0.5}$-*score*s.

A.3 Sampling Density

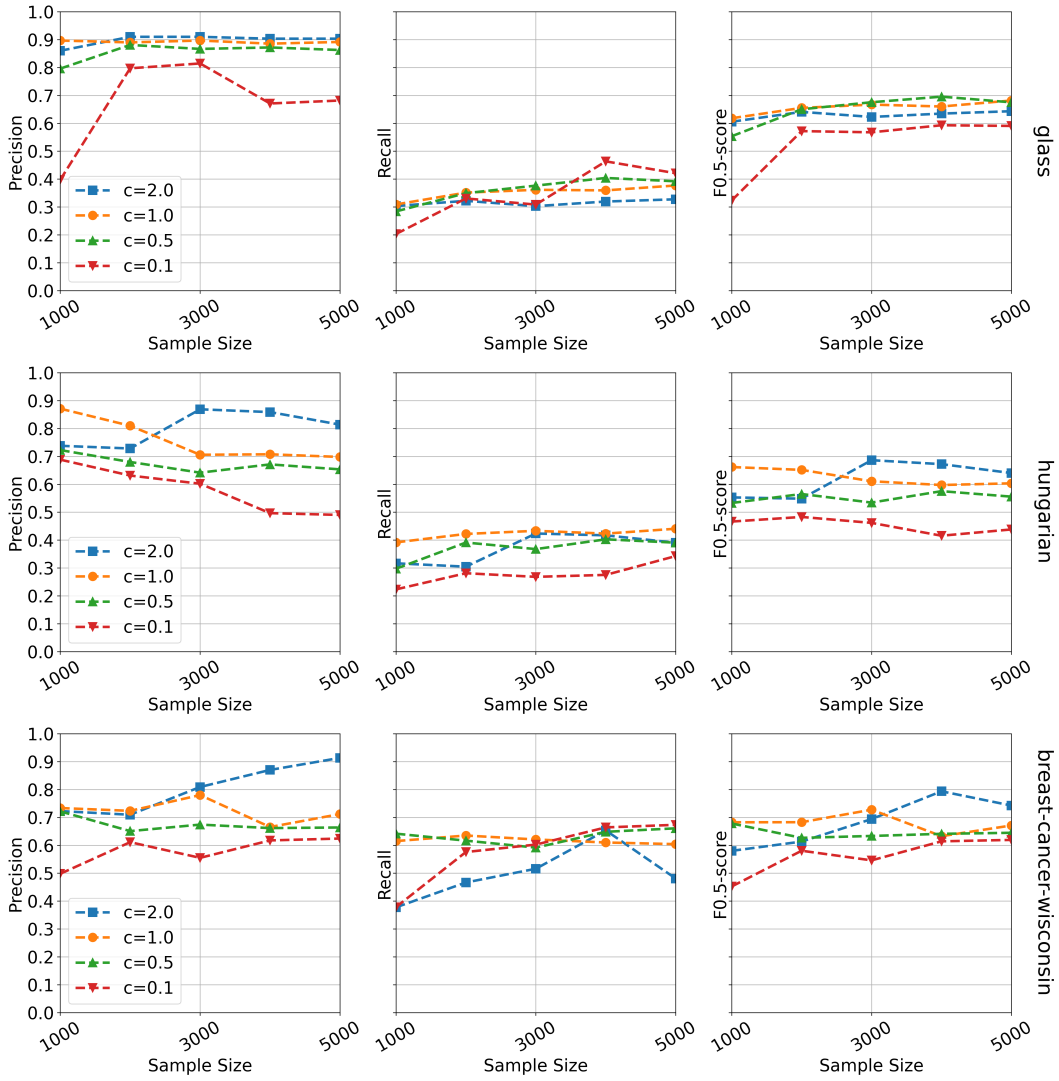

Figure A.3: Various coefficients for sampling distribution can result in different explanation accuracies.

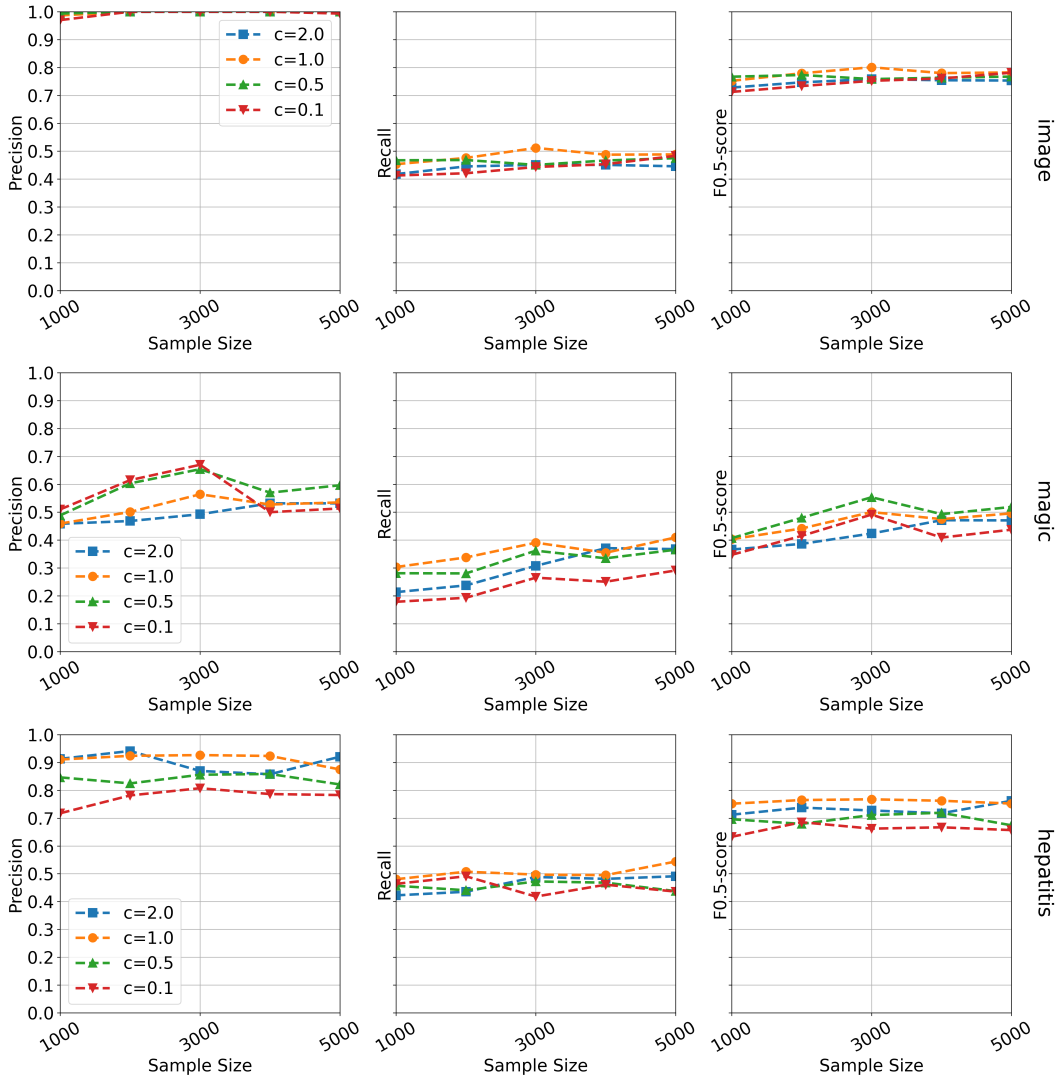

Figure A.3: Various coefficients for sampling distribution can result in different explanation accuracies.

A.4 Multi-class vs Binary Neighbourhoods

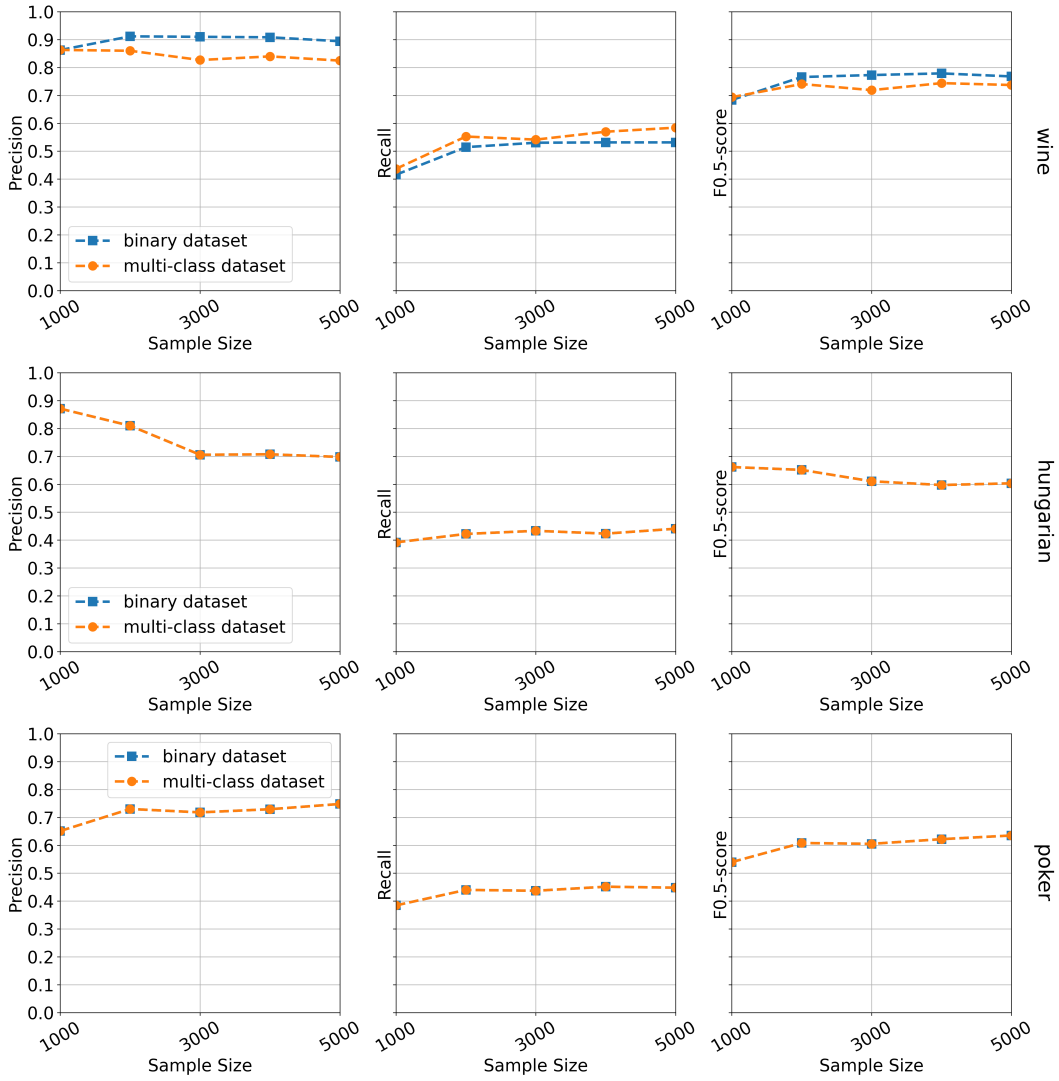

Figure A.4: The effect of keeping multi-class datasets as they are against transforming them into a "One-vs-Rest" dataset.

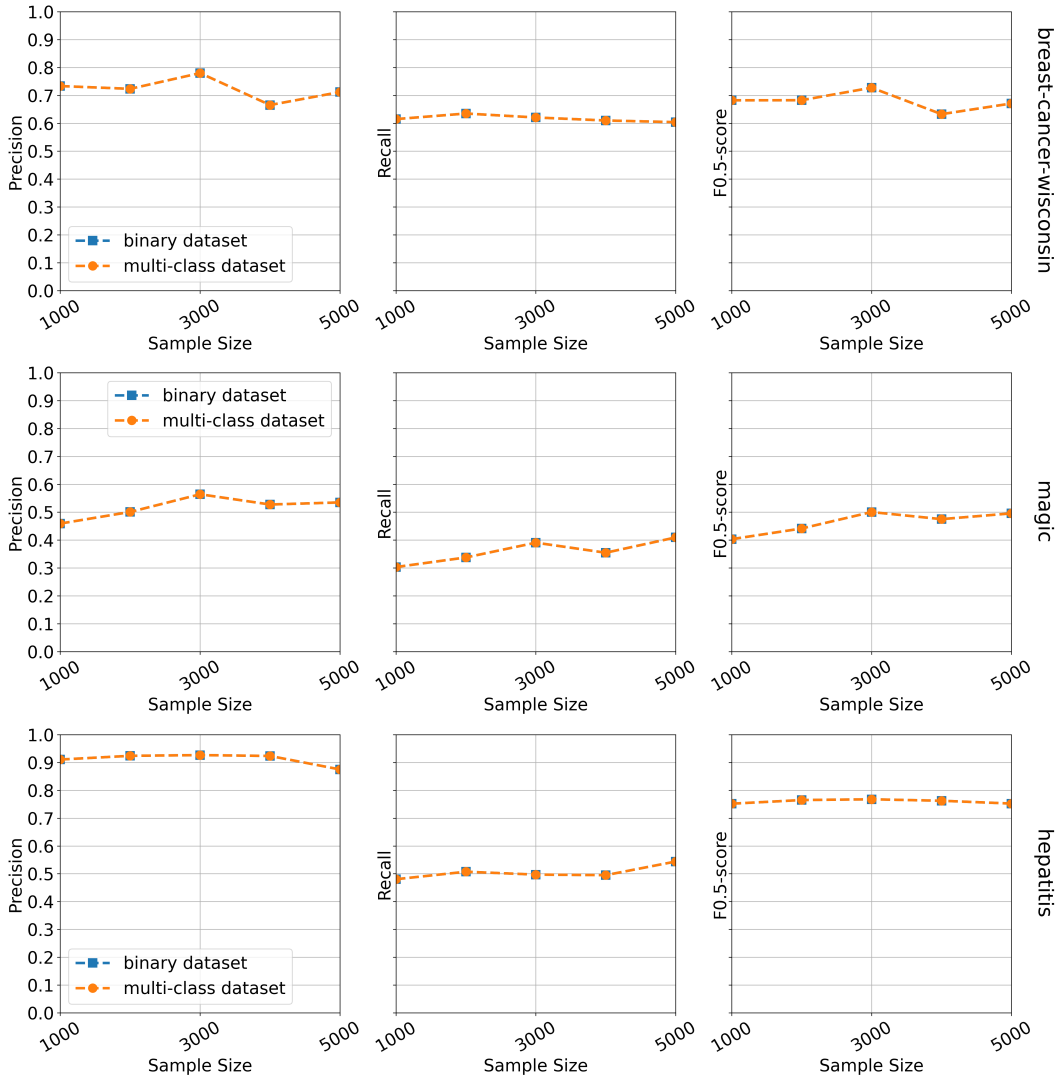

Figure A.4: The effect of keeping multi-class datasets as they are against transforming them into a "One-vs-Rest" dataset.

A.5 Rule Selection

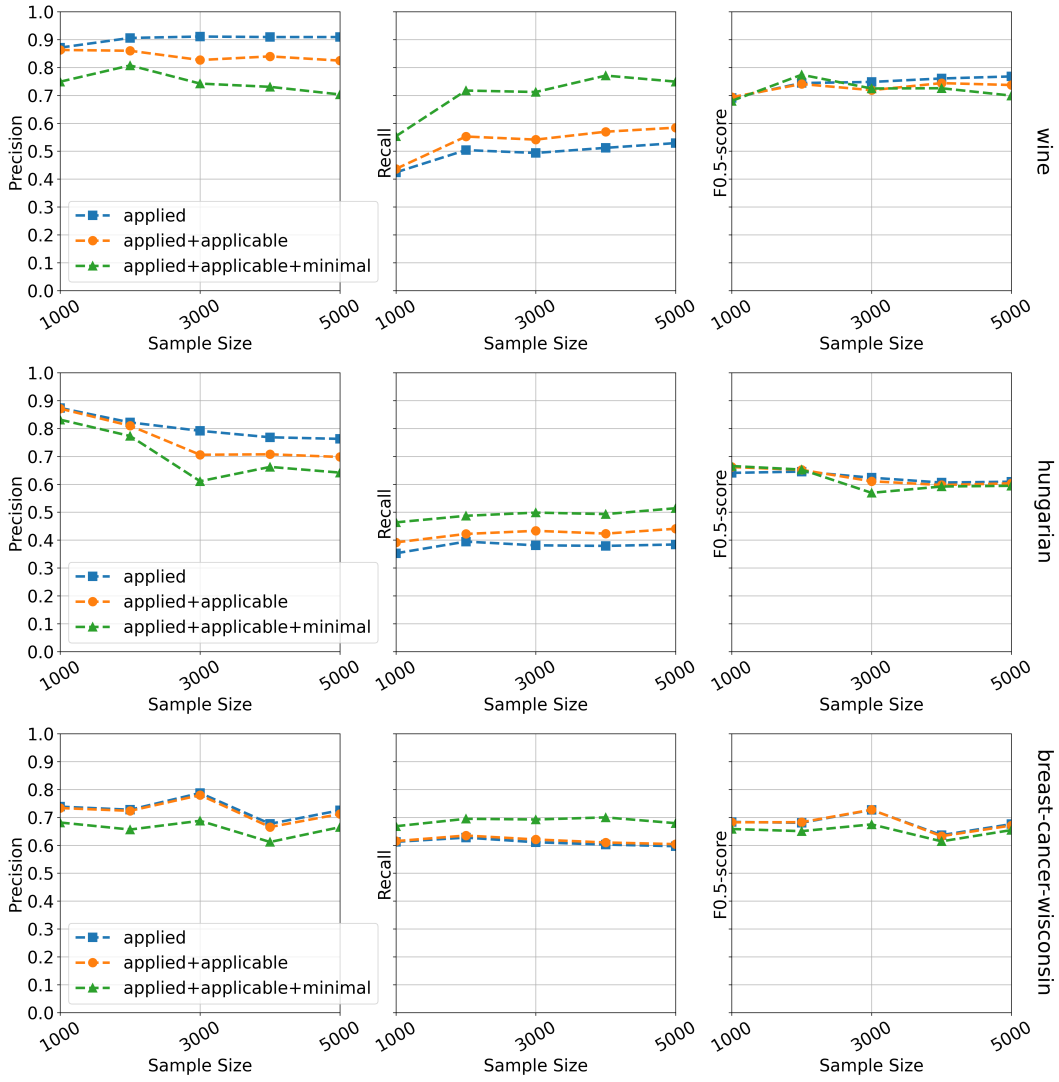

Figure A.5: Different rule selection criteria result in different *Precision*, *Recall*, and $F_{0.5}$-*score*.

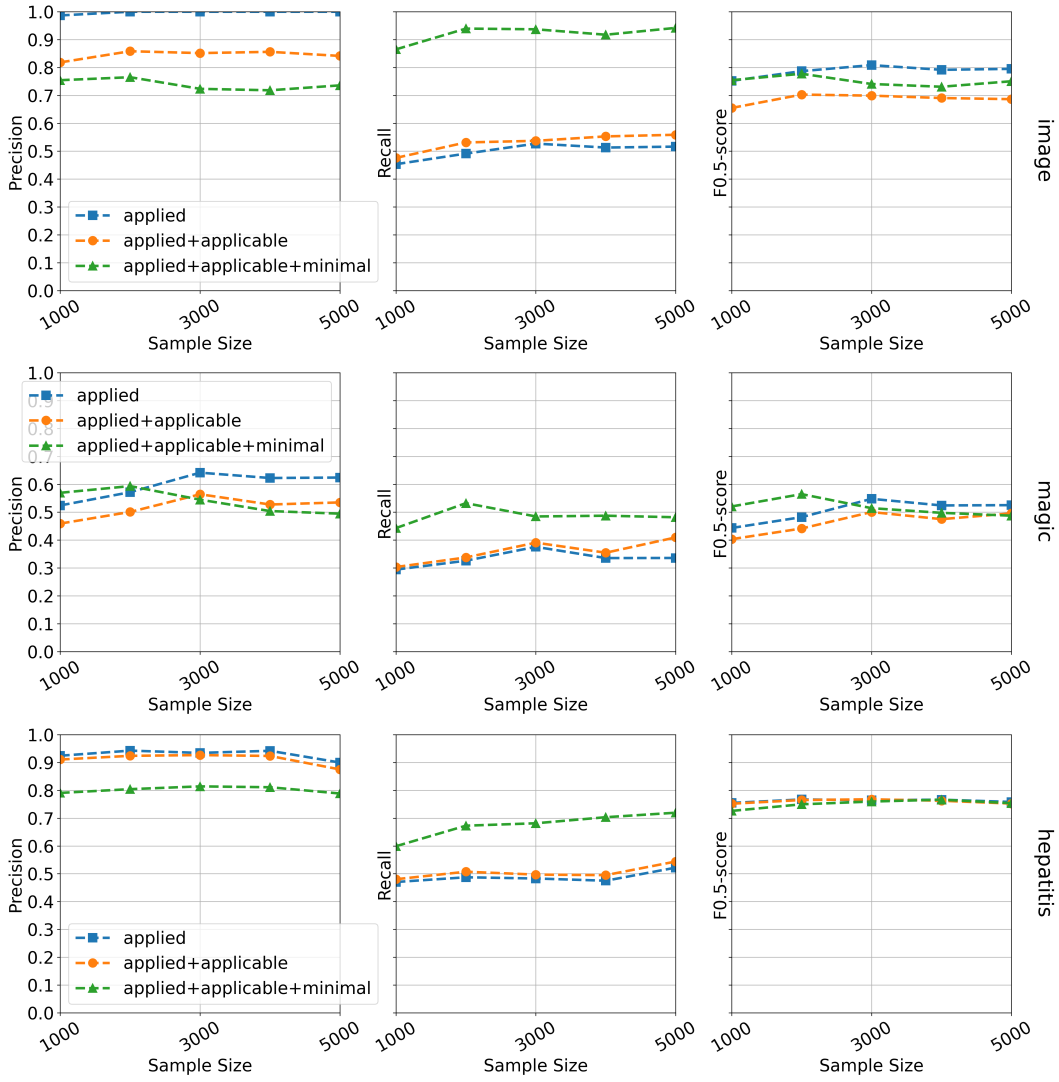

Figure A.5: Different rule selection criteria result in different *Precision*, *Recall*, and $F_{0.5}$-*score*.

A.6 Feature Extraction from Rules

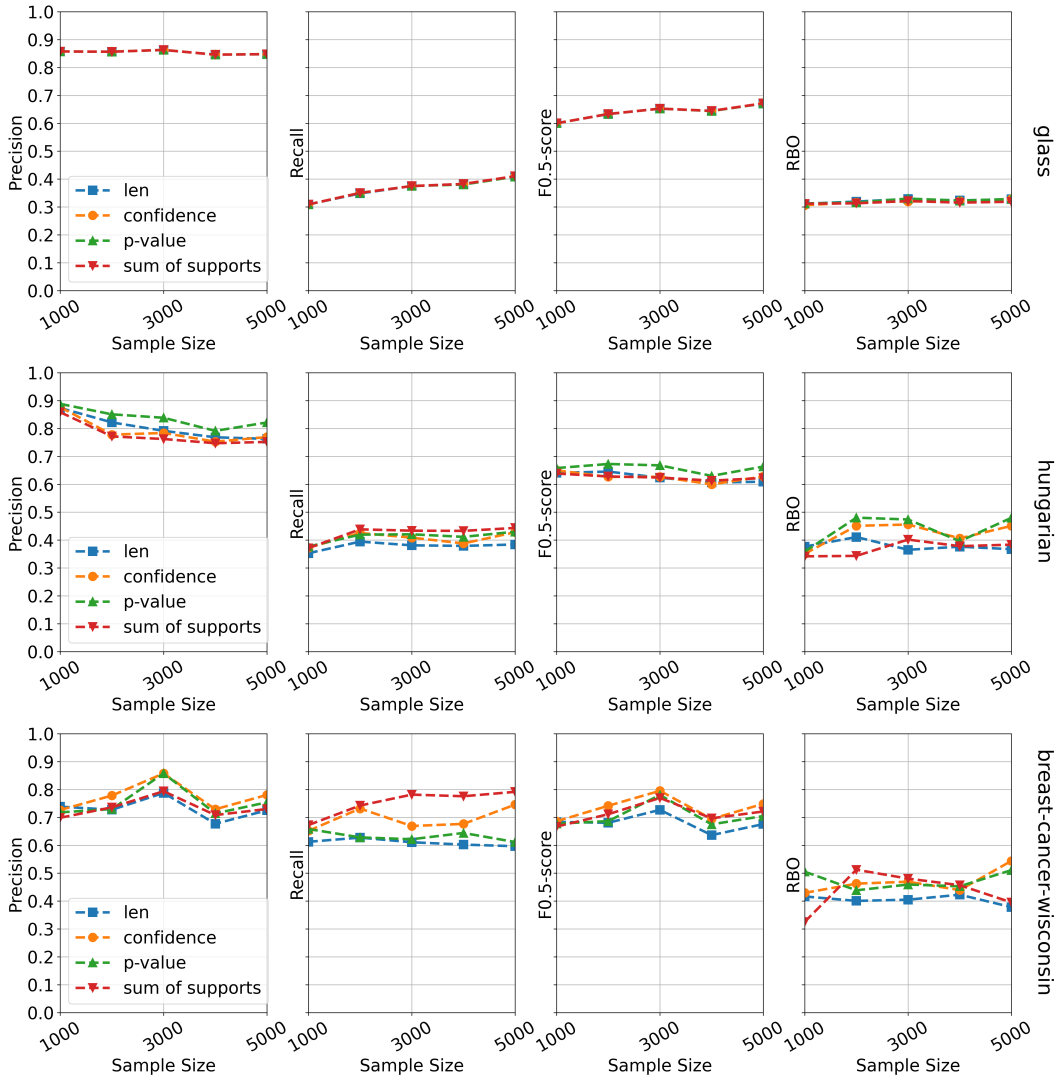

Figure A.6: Different feature selection criteria result in different *Precision*, *Recall*, $F_{0.5}$, and *RBO* scores.

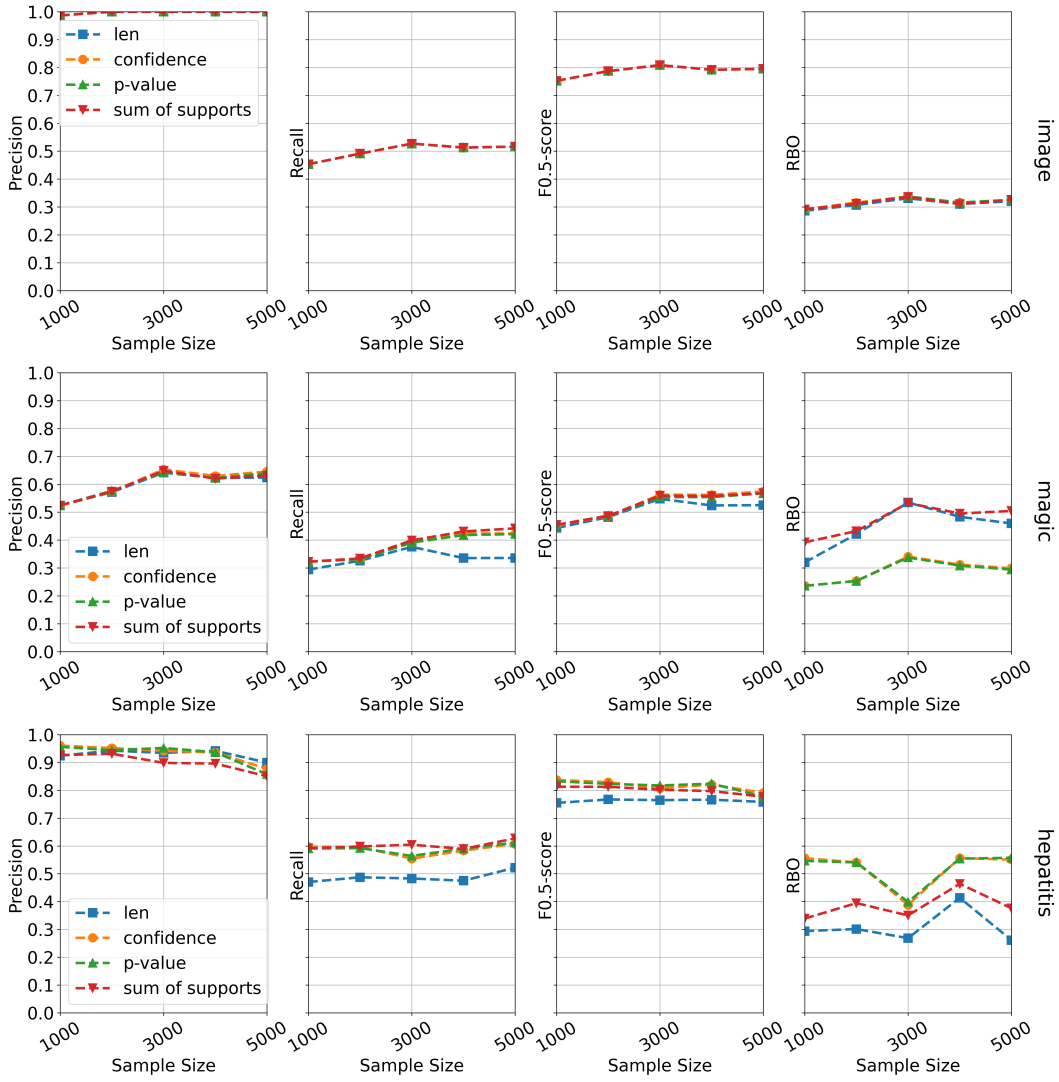

Figure A.6: Different feature selection criteria result in different *Precision*, *Recall*, $F_{0.5}$, and *RBO* scores.

A.7 Comparison against other methods

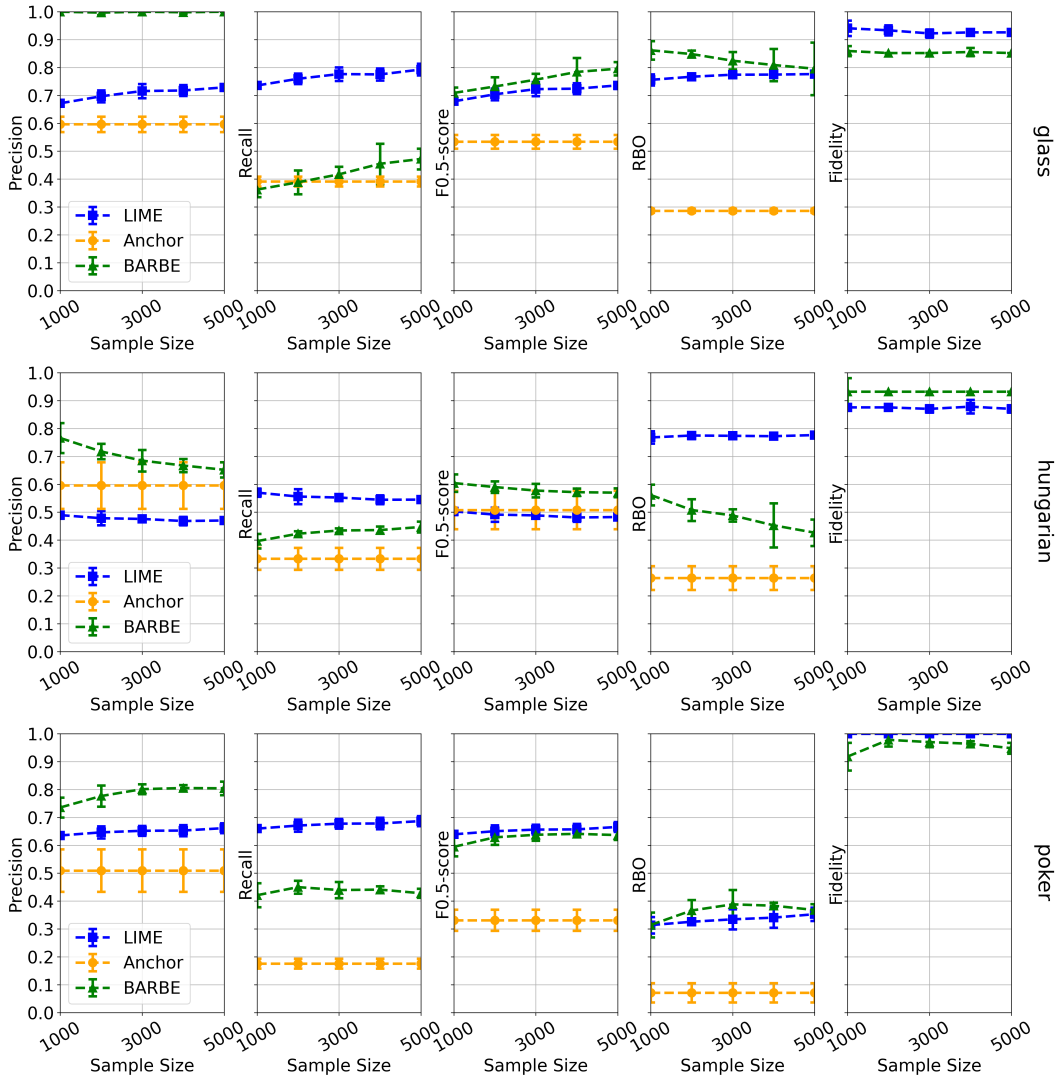

Figure A.7: Performance of LIME, Anchor, and BARBE in different datasets.

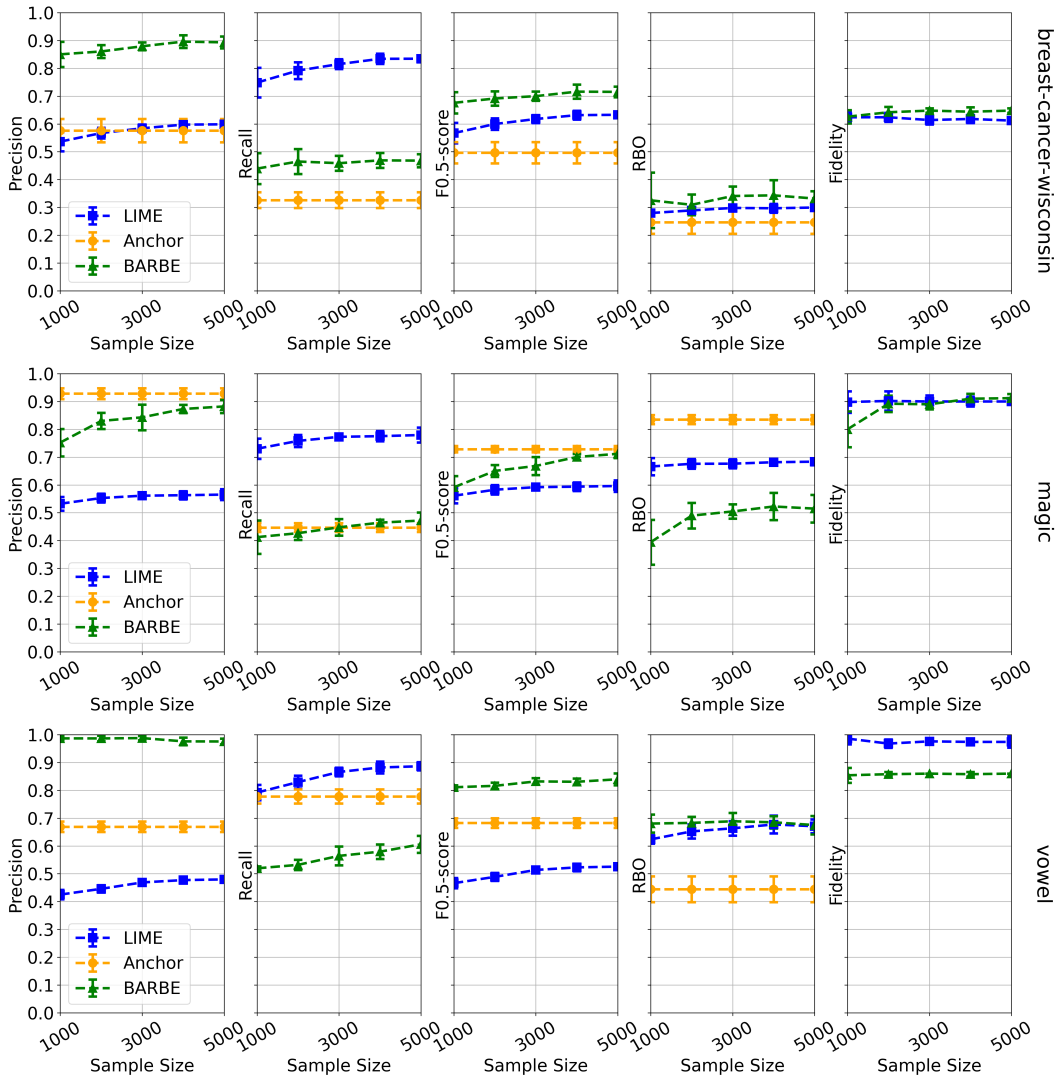

Figure A.7: Performance of LIME, Anchor, and BARBE in different datasets.

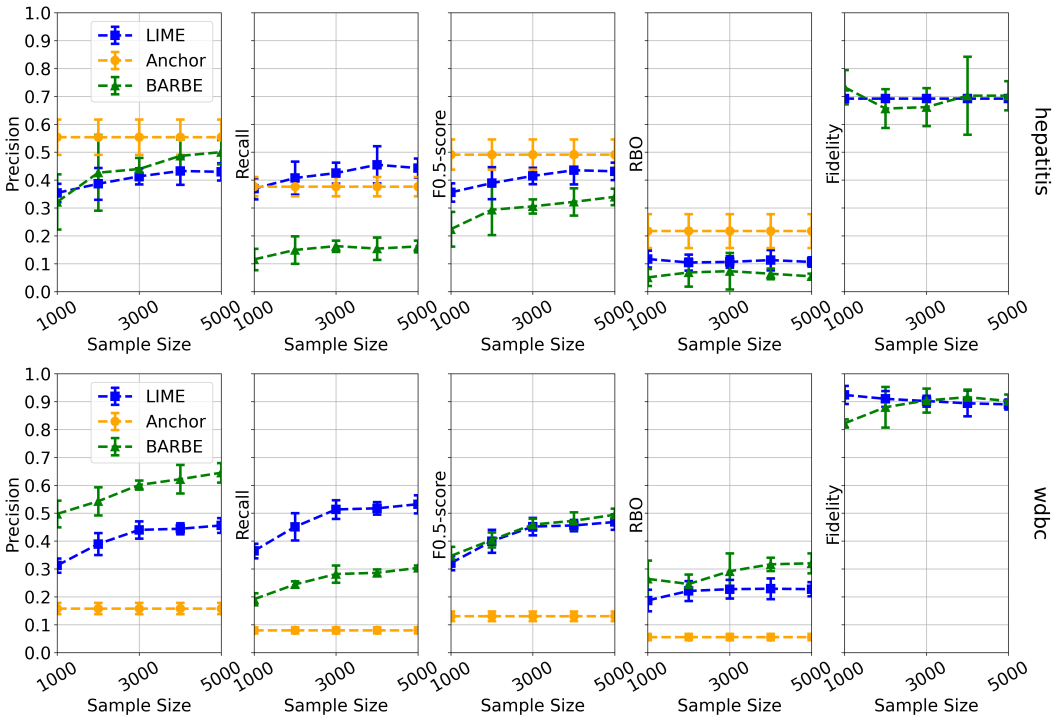

Figure A.7: Performance of LIME, Anchor, and BARBE in different datasets.