UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Renato Miranda Filho

Explaining Regression Models Predictions

Belo Horizonte
2023

Renato Miranda Filho

Explaining Regression Models Predictions

Final Version

Dissertation presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Doctor in Computer Science.

Advisor: Gisele Lobo Pappa

Belo Horizonte
2023

Miranda Filho, Renato.

M672e Explaining regression models predictions [recurso eletrônico] / Renato Miranda Filho – 2023.
1 recurso online (193 f. il, color.) : pdf.

Orientadora: Gisele Lobo Pappa.
Tese (Doutorado) - Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação.
Referências: f. 149 -160.

1. Computação – Teses. 2. Aprendizado de máquina – Teses. 3. Análise de regressão – Modelos computacionais – Teses. I. Pappa, Gisele Lobo. II. Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciência da Computação. III. Título.

CDU 519.6*82(043)

Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa
CRB 6/1510 Universidade Federal de Minas Gerais - ICEx

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

EXPLAINING REGRESSION MODELS PREDICTIONS

RENATO MIRANDA FILHO

Tese defendida e aprovada pela banca examinadora constituída pelos Senhores(a):

Profa. Gisele Lobo Pappa - Orientadora
Departamento de Ciência da Computação - UFMG

Prof. Renato Vimieiro
Departamento de Ciência da Computação - UFMG

Prof. Wagner Meira Júnior
Departamento de Ciência da Computação - UFMG

Prof. Fabrício Olivetti de França
Centro de Matemática Computação e Cognição - Universidade Federal do ABC

Prof. Frederico Gadelha Guimarães
Departamento de Engenharia Elétrica - UFMG

Prof. Ricardo Cerri
Departamento de Computação - UFSCar

Belo Horizonte, 24 de abril de 2023.

Documento assinado eletronicamente por **Gisele Lobo Pappa**, **Professora do Magistério Superior**, em 26/04/2023, às 10:55, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Wagner Meira Junior**, **Professor do Magistério Superior**, em 27/04/2023, às 18:32, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Renato Vimieiro**, **Professor do Magistério Superior**, em 28/04/2023, às 08:52, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Frederico Gadelha Guimaraes**, **Professor do Magistério Superior**, em 28/04/2023, às 13:51, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Ricardo Cerri**, **Usuário Externo**, em 13/06/2023, às 16:23, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Fabricio Olivetti de França**, **Usuário Externo**, em 15/06/2023, às 08:26, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2246203** e o código CRC **ED9D495A**.

Acknowledgments

I thank God for everything!

I would like to express my sincere gratitude to all those who supported me during the challenging journey of completing my doctoral degree, particularly amidst the pandemic.

Firstly, I want to express my heartfelt thanks to my advisor, Gisele Lobo Pappa, for her guidance, encouragement, and unwavering support throughout my research. Her expertise and mentorship were instrumental in shaping my work, and I am grateful for the opportunity to have learned from her.

I would also like to extend my gratitude to my parents, Renato and Sonia, for their constant love, support, and belief in me. To my sister Ana, my brother-in-law Jaurés, and my nephews Gabriel, Isaac, and Pedro, I am grateful for their encouragement and understanding, especially during the difficult moments of this journey.

I would like to thank the members of my doctoral committee, Fabricio Olivetti, Frederico Gadelha, Renato Vimieiro, Ricardo Cerri, and Wagner Meira, for their invaluable feedback, constructive criticism, and support throughout the process. Their insights and expertise have helped me to refine and improve my work.

Lastly, I am grateful to the Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais (IFMG) – Campus Sabará for their support and resources that allowed me to pursue my research.

"*Knowledge is a process of piling up facts;*
wisdom lies in their simplification."
(Martin H. Fischer)

Resumo

O crescente interesse em aplicações de aprendizado de máquina levantou uma discussão na comunidade de inteligência artificial sobre a transparência do modelo. No centro desta discussão estão questões sobre a explicação e a interpretabilidade do modelo. Embora alguns métodos sejam sistematicamente apontados como interpretáveis por humanos, por exemplo, Programação Genética (PG) e Árvores de Decisão (AD), sabemos que quanto mais complexo o modelo se torna, menos interpretável ele é. Esta tese enfoca na explicabilidade dos modelos de regressão. As abordagens propostas não explicam o processo seguido por um modelo para se chegar a uma decisão. Em vez disso, elas justificam as previsões que o modelo faz. Quatro métodos agnósticos de modelo foram propostos para ajudar a justificar as previsões: dois métodos baseados em atributos, denominados Explicação por Aproximação Local (ELA) e Explicação Dinâmica por Aproximação Local (DELA), um método baseado em protótipos, denominado Explicação Multiobjetivo Baseada em Protótipos para Regressão (M-PEER) e um método híbrido multinível denominado Explicação Multinível Híbrida (HuMiE). ELA é um método de explicação simples que encontra os vizinhos mais próximos da instância que queremos explicar e realiza uma regressão linear usando este subconjunto de instâncias. Os coeficientes dessa regressão linear são então usados para gerar uma explicação local para o modelo. Os resultados mostram que os erros obtidos pelo método ELA são semelhantes aos da regressão realizada com todos as instâncias. Além disso, propomos o uso de visualizações simples baseadas em gráficos de áreas empilhadas para mostrar as explicações do método ELA. Assim, observamos que os resultados obtidos resultam em informações adicionais aos usuários sobre os atributos mais relevantes para as previsões. DELA é um método que aprimora ELA deixando-o com uma estrutura interna mais dinâmica e adaptável às características do conjunto de dados trabalhado. Para tanto, DELA realiza a normalização dos conjuntos de dados de entrada, escolhe a métrica de distância mais adequada em cada contexto ao comparar as instâncias, define dinamicamente o número de vizinhos que serão selecionados para a explicação local de acordo com a densidade local e calcula a importância dos recursos com base na localização da instância de teste. Desta forma, DELA demonstrou ser menos afetado por conjuntos de dados com altas variâncias nos atributos, além de retornar interpretações semanticamente corretas, o que nem sempre ocorre nos métodos explicativos concorrentes. O terceiro método proposto, M-PEER, utiliza protótipos para fornecer explicações globais e locais para problemas de regressão. Este método leva em consideração os recursos de entrada e também a saída do modelo.

Definimos protótipos como casos exemplares no domínio do problema. O M-PEER é um método baseado em um algoritmo evolutivo multiobjetivo que minimiza o erro do modelo explicável e também outras duas medidas semanticamente baseadas no contexto de interpretabilidade. Os resultados mostram ganhos significativos do M-PEER sobre outras estratégias, com uma melhoria média de 12% em uma métrica proposta denominada Fidelidade e Estabilidade Global (GFS) e 17% na Raiz do Erro Quadrático Médio (RMSE) quando comparado com ProtoDash, um método considerado estado-da-arte em explicações baseadas em protótipos. Por fim, o método denominado HuMiE é capaz de criar explicações híbridas multinível para problemas de regressão. Por híbrido entendemos que a explicação fornecida engloba tantos elementos de uma explicação baseada em exemplos quanto de explicações baseadas em atributos. Por multinível HuMiE apresenta em formato de árvore as explicações globais do modelo, explicações locais para instâncias de teste específicas e também explicações intermediárias, retratando subgrupos semanticamente semelhantes. Experimentos com conjuntos de dados do mundo real mostraram quantitativamente que os protótipos escolhidos por HuMiE foram melhores que todos os concorrentes (M-PEER e ProtoDash) em relação à métrica de fidelidade. HuMiE também foi melhor que M-PEER com um único nível em relação às métricas de estabilidade e GFS. Qualitativamente HuMiE é capaz de diversificar na escolha de protótipos de acordo com as características do conjunto de dados apresentado, tanto em termos de saída do modelo quanto dos atributos. Além disso, HuMiE foi capaz de encontrar subgrupos de instâncias semelhantes, proporcionando uma interpretação intermediária entre as escalas local e global.

Palavras-chave: regressão; explicação; interpretabilidade.

Abstract

The growing interest in machine learning applications has raised a discussion in the artificial intelligence community about model transparency. In the center of this discussion is the question of model explanation and interpretability. Although some methods are systematically pointed out as interpretable by humans, e.g., genetic programming (GP) and Decision Trees (DT), we know that the more complex the model becomes, the less interpretable it is. This thesis focuses on explainability of regression model. The proposed approaches do not explain the process followed by a model to reach a decision. Instead, they justify the predictions the model makes. Four model-agnostic methods were proposed to help justifying predictions: two features-based methods, called Local Approximation Explanation (ELA) and Dynamic Explanation by Local Approximation (DELA), a prototype-based method, called Multiobjective Prototype-basEd Explanation for Regression (M-PEER) and a multilevel hybrid method called Hybrid Multilevel Explanation (HuMiE). ELA is a simple explanation method that finds the nearest neighbors of the instance we want to explain and performs a linear regression using this subset of instances. The coefficients of this linear regression are then used to generate a local explanation to the model. Results show that the errors of ELA are similar to those of the regression performed with all instances. Furthermore, we propose the use of simple visualizations based on stacked area plots to show explanations of the ELA method. Thus, we observed that the results obtained can give many insights to the users about the most relevant features to the predictions. DELA is a method that improves ELA, leaving it with a more dynamic and adaptable internal structure to the characteristics of the dataset worked on. To do so, DELA normalizes the input datasets, chooses the most appropriate distance metric in each context when comparing the instances, dynamically defines the number of neighbors that will be selected for the local explanation according to the local density and calculates the importance of features based on the location of the test instance. In this way, DELA proved to be less affected by datasets with high variances in the features, in addition to returning semantically correct interpretations, which does not always occur in competing explanatory methods. The third proposed method, M-PEER, uses prototypes to provide global and local explanations for regression problems. This method takes into account the input features as well as the output of the model. We define prototypes as exemplary cases in the problem domain. M-PEER is based on a multi-objective evolutionary algorithm that optimizes both the error of the explainable model and two other semanticall-based measures of interpretability. The results show significant gains of the

M-PEER over other strategies, with an average improvement of 12% in a proposed metric called Global Fidelity and Stability (GFS) and 17% in Root Mean Squared Error (RMSE) when compared to ProtoDash, a state-of-the-art method in prototype-based explanations. Finally, the method called HuMiE is capable of creating multilevel hybrid explanations for regression problems. By hybrid, we mean that the explanation provided encompasses as many elements of an example-based explanation as it does of features-based explanations. By multilevel HuMiE presents in a tree format the model's global explanations, local explanations for specific test instances and also intermediate explanations, portraying semantically similar subgroups. Experiments with real-world datasets quantitatively showed that the prototypes chosen by HuMiE outperformed all competitors (M-PEER and ProtoDash) on the fidelity metric. HuMiE was also better than M-PEER with a single level regarding stability and GFS metrics. Qualitatively HuMiE is able to diversify in the choice of prototypes according to the characteristics of the presented dataset, both in terms of model output and features. Furthermore, HuMiE was able to find subgroups of similar instances, providing an intermediate interpretation between local and global scales.

Keywords: regression; explanation; interpretability.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

Machine learning models are increasingly being used to make critical decisions. As Aristotle once said, "we do not think we understand something until we have grasped the why of it" [72]. This same reasoning applies to most machine learning models widely used today. In many cases, these models are able to create good generalizations by predicting outputs for unprecedented events, even though we are not able to understand why or how their internal mechanisms work and, consequently, why a specific output was obtained. Hence, preeminent models are often opaque and difficult to interpret, and because of that we call these models black-boxes.

However, as the most accurate models are usually black-box models, works focusing on model explainability and interpretability have grown in importance [68, 49]. Explainable/Interpretable models are especially important in domains where decisions lead to critical consequences. For instance, the use of black-box machine learning models has already led to unfairly long prison sentences (e.g., prisoner Glen Rodriguez had his parole denied due to an incorrect COMPAS [1] score) [116].

Note that there is a difference between interpretable and explainable models. Interpretable models are those that are inherently understandable by human beings, such as linear models, decision rules, decision trees and logistic regression. But even interpretable models can be difficult to understand by a human, and there many possible causes for that: the high dimensionality of the data, the large volume of instances to be analyzed, complex mathematical functions returned, big sets of rules or extensive trees.

Explanable models, on the other hand, are obtained by elucidating a black-box model in order to make its internal mechanisms, which led to a given prediction, minimally comprehensible to humans [101, 62]. Most research on explainable models for machine learning refers to classification tasks [106, 108, 99, 110, 41, 103, 52, 85, 14, 112, 26, 65, 96]. However, some initiatives have already been proposed to explain regression models [87, 7, 62, 101, 51].

Briefly summarizing, classification problems consist in learning a model that expresses the relationships between a set of features that describe a given instance and a

[1] COMPAS is a case management and decision support proprietary system used by U.S. courts to assess the likelihood of a defendant becoming a recidivist.

fixed predefined set of classes. Thus, the goal is to predict the correct class of instances not previously labeled. In regression problems, in contrast, the output is continuous, leading to an unlimited number of possible values to be predicted by the model. In this thesis we will focus on the task of providing explanations for regression problems.

Two more popular approaches used by explanation methods are: feature-based and example-based [82]. Feature-based approaches analyze data characteristics – including, for example, statistical summaries and importance of features – that make a particular black-box model perform a predict.

In contrast, example-based or prototype-based approaches select or create instances that can represent the black-box model or a specific test instance prediction. We call these explanatory instances *prototypes*. The idea behind finding the best prototypes to explain predictions is in line with the mode of human reasoning that seeks to find cases similar to a certain phenomenon when trying to understand it, and is crucial to construct effective strategies for tactical decision-making [1, 104]. However, isolated prototypes are not enough to explain predictions. The set of selected prototypes needs to be able to both describe the data distribution considering the fitted machine learning model while also explaining individual predictions.

This thesis investigates the problem of making the internal behavior of black-box regression models understandable to humans using explanation methods. To achieve that, we propose feature-based and prototype-based methods, as well as a new hybrid multi-level approach that combines elements of the two previously mentioned approaches to provide an hierarchical understanding of the model developed. Our hypothesis is that we can improve the quality of explainability of black-box regression models. By quality we mean that the explanations provided should be consistent with the domain expert knowledge and, at the same time, capable of minimizing error metrics in relation to the model and the instances to be explained. Finally, the proposed methods should be intrinsically simple, preferably using elements that can be extracted from the model's own training set. This makes explanations more accessible and easy to understand, even for users with little technical knowledge. In summary, this thesis contributes to advancing the understanding of black-box models and may have important implications in different areas of everyday life where regression models are used.

1.1 Motivation

As stated before, understanding the reasons certain regression models make specific predictions is crucial in application problems that can directly affect individuals. Better

understanding of the models can help: i) developers to improve the results obtained by a model [76] by, for example, identifying resources that should not be used in the prediction; ii) users to gain insights on how to improve their services or methods [76, 62]; and iiii) assess fairness and privacy [30]. Furthermore, explanations may be required by law [62, 122, 44] for individuals to obtain appropriate feedback of results obtained by models who used their personal data.

Much has been done in this area in recent years [88, 49, 18, 76]. Despite that, there are still many open questions regarding the methods of explanation, especially when working with regression problems. Below we detail four of these open issues that will be investigated in this thesis:

Issue 1: In general, explanation methods were not designed and evaluated specifically for the context of regression.

The main focus of some of the current explanation methods is the classification task, requiring some adaptations to work in regression. For example, methods for feature-based explanations need adaptations to be used in regression problems or even discretize the regressor output in classes to be used for explanation in regression models, such as LACE [97], Anchors [108], and CEC [27]. Other works did not use datasets of regression problems (LIME [105]) or used only a small regression set in their tests (MAME [101]), making it difficult to provide conclusive results in this scenario. We also observe works with prototypes-based explanations that do not use the continuous output provided by the regressor to obtain the most adequate explanations (PS [14], MMD-critic [68] and ProtoDash [51]), i.e., they end up focusing their explanations on the original distribution of the data and not on the evaluated model.

Issue 2: The local explanation is not always coherent with the global explanation provided by the same explanation method.

Explaining the output given to a specific instance is considerably simpler than given an overall explanation of the model, since it only requires understanding the local behavior of a model. A valid analysis for this task would be to place a magnifying glass in a specific region of the model's solution space. Some proposed methods perform this task, for example, by creating perturbation instances from the instance of interest [107, 62, 122], creating fictitious instances through genetic algorithms [48], while others analyze the local distribution of data [68, 51]. However, generalizing this local observation to a global explanation of the entire model and, at the same time, preserving its local characteristics is not a trivial task. When trying to do that we have the risk that the global explanation provided will be as difficult to understand as the initial model. For example, if the global model explanation turns out to be a complex equation, we are actually replacing a black-box regression model by a black-box explanation also incomprehensible to humans.

Issue 3: Semantically invalid instances for the application can be created by locally explaining the model.

Usually, the algorithms that deal with local explanation create a perturbation around samples of interest in order to create fictitious samples and then describe the local behavior [107, 62, 122]: this can generate semantically invalid instances that would simply not be observed in the real world.

Issue 4: Metrics to measure the similarity of instances and to evaluate the explanatory power of methods are little explored, and significantly impact the results of explanations.

Most current explanatory methods need, at some point, to use similarity metrics to compare different instances. The most widely used metric is the Euclidean distance when evaluating tabular dataset [105, 101, 62, 48] . However, the misuse of the Euclidean distance can generate not faithful comparisons of pairs of instances, especially when dealing with high dimensional data in terms of the number of features, leading to the problem of the curse of the dimensionality [29, 4].

In addition, when evaluating different methods of explanation, it is common to use only one error-based metric [14, 68, 51, 52, 48, 87], which we consider is not always appropriate. The use of metrics that can qualitatively assess the quality of the explanations can be considered the holy grail of this area, which still heavily relies on qualitative and expensive analysis from humans.

1.2 Objectives

The main objective of this thesis is to provide mechanisms for a user to understand the internal reasons that drive the behavior of a given black-box regression model, i.e., the factors that led to a certain predict. In order to achieve that, we pose the following research questions:

Research Question 1 (RQ1): Is it possible to generalize local explanations learned from black box regression models in order to create globally coherent and semantically valid explanations when compared to those provided by domain knowledge specialists? (Relates to Issue 2)

Research Question 2 (RQ2): Is it really necessary to create fictitious instances in order to provide a local explanation of a given instance? (Relates to Issue 3)

Research Question 3 (RQ3): Which distance metric performs better, especially when working with high-dimensional data, when used to measure the similarity of instances in regression problems in order to provide explanations? (Relates to Issue 4)

Research Question 4 (RQ4): How does explanation quality – measured by error – improve when we consider not only the distribution of datasets in regression problems but also the continuous nature of the outputs from regression problems? (Relates to Issue 1)

Research Question 5 (RQ5): What evaluation measures can be used to assess the quality of an explanation beyond the error? (Relates to Issue 4)

Research Question 6 (RQ6): Can we create an intermediate level of explanations that bridges the gap between global and local explanations in regression models? (Relates to Issue 2)

1.3 Publications

This thesis has resulted in the following publications so far:

- R. Miranda Filho, A. M. Lacerda and G. L. Pappa, Explaining Symbolic Regression Predictions, 2020 IEEE Congress on Evolutionary Computation (CEC), 2020, pp. 1-8, doi: 10.1109/CEC48606.2020.9185683.
- R. Miranda Filho, A. M. Lacerda, and G. L. Pappa. 2023. Explainable Regression Via Prototypes. ACM Trans. Evol. Learn. Optim. 2, 4, Article 14 (December 2022), 26 pages. https://doi.org/10.1145/3576903.

During the doctoral program, we also collaborated with other professors and students, generating the following publications related to the field of Machine Learning.

- Pedro V. Brum, Matheus C. Teixeira, Renato Miranda, Renato Vimieiro, Wagner Meira Jr, and Gisele L. Pappa. 2020. A Characterization of Portuguese Tweets Regarding the Covid-19 Pandemic. In Anais do VIII Symposium on Knowledge Discovery, Mining and Learning, outubro 20, 2020, Evento Online, Brasil. SBC, Porto Alegre, Brasil, 177-184. DOI: https://doi.org/10.5753/kdmile.2020.11974.

In addition, we also contribute to the elaboration of the platform LEMONADE (Live Exploration and Mining Of a Non-trivial Amount of Data from Everywhere) documentation.

Another paper was submitted and is currently under review:

- R. Miranda Filho, G. S. I. Aldeia, F. O. França and G. L. Pappa, Understanding explanations in regression: What can state-of-the-art explanatory methods say about a model?. Machine Learning. 2023.

Finally, two other papers are completed and ready for submission. The first one is about our proposal to improve feature-based explanation (DELA), and the other is about our proposal for a hybrid multi-level explanation (HuMiE).

1.4 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 presents the fundamental concepts and related works on methods for explaining regression models. Chapter 3 presents ELA, a method that we proposed for explanation feature-based and that partially answers questions RQ1, RQ2 and RQ3. Next, Chapter 4, studies improvements that can be made in explanation methods to gain a deeper understanding of RQ3 and provide a direct response to RQ4. Additionally, we introduce the second version of the feature-based explanation method (DELA). Chapter 5 presents a third method, M-PEERs, that follows an explanation approach prototype-based, which directly address questions RQ5 and RQ6. Chapter 6 presents a fourth method, HuMiE, which provides a hybrid explanation multilevel, composed of feature-based elements and prototype-based elements, which directly address questions RQ1 and RQ6. Finally, Chapter 7 draws conclusions and points out possibilities for future work.

Chapter 2

Related Work

The use of machine learning models is already common in our daily lives. Several tasks are performed with models produced by these techniques including, for example, predicting the credit score for an individual, selecting job interview candidates or recommending movies. In this context, understanding the results provided by machine learning models has become a very relevant problem in the literature, specially after many cases pointed out to the danger associated with automatic decisions made by models [88, 49, 18, 120].

Machine learning models are usually divided into two broad categories: intrinsically interpretable and black-box models. Intrinsically interpretable models (or white-box models) are those based on simple structures — including linear regression, decision trees, decision rules, logistic regression, or kmedoids — that humans can understand. Note that even models considered interpretable can have internal behaviors extremely difficult to understand. Some examples of how difficult the human understanding of these models can be: complex mathematical expressions with many variables (describing the different features of the problem), very deep decision trees or with many branches, extensive and complex rules, and a large amount of considered instances relevant.

In contrast, black-box models work based on complex internal mechanisms mostly difficult for humans to understand. They include models such as artificial neural networks, support vector machines, and ensembles of different kinds [75].

Black-box models often generate the best results in terms of accuracy for classification and regression problems [49]. However, in many situations, the need to understand how decisions have been made by a model — to convince a specialist to trust the model or to comply with current regulations to explain algorithm decisions [64] — makes these models not the most appropriate. To be able to better understand decisions made by black-box models, explanation methods were proposed.

Most explanation methods are built after generating the black-box model and produce an auxiliary tool to help humans understand the outputs of the original black-box model. While there has been extensive literature on explanation methods for classification, there is still a lack of studies on how these methods might help in regression tasks [73]. For many, there are still difficulties in understanding why we even need to explain regression models.

Regression analysis methods aim to model the associations between a set of variables and an observed outcome. They can be used for *(i)* prediction — when, given the values of the predictions, we want to interpolate or extrapolate from the observations to estimate the outcome, *(ii)* to find associations between the independent variables and the outcome and how strong these associations are, or *(iii)* to find causal relations — given that the independent variables cause the outcome, one can check how much change a variable causes to the outcome given a unitary change.

Nevertheless, a regression model should not only make predictions. It may also be desirable that the model informs us about the shape and strength of these associations. Because of that, linear regression and the generalized linear model are often the choices of practitioners to make this type of analysis. Their main advantage is that they provide a straightforward value to the extent of the association strength of a specific variable.

However, non-linear models are required in many scenarios to capture genuine associations between variables and outcomes. In this case, understanding these associations and their strength depends on understanding a complex model, in which explanation methods become crucial in prediction tasks.

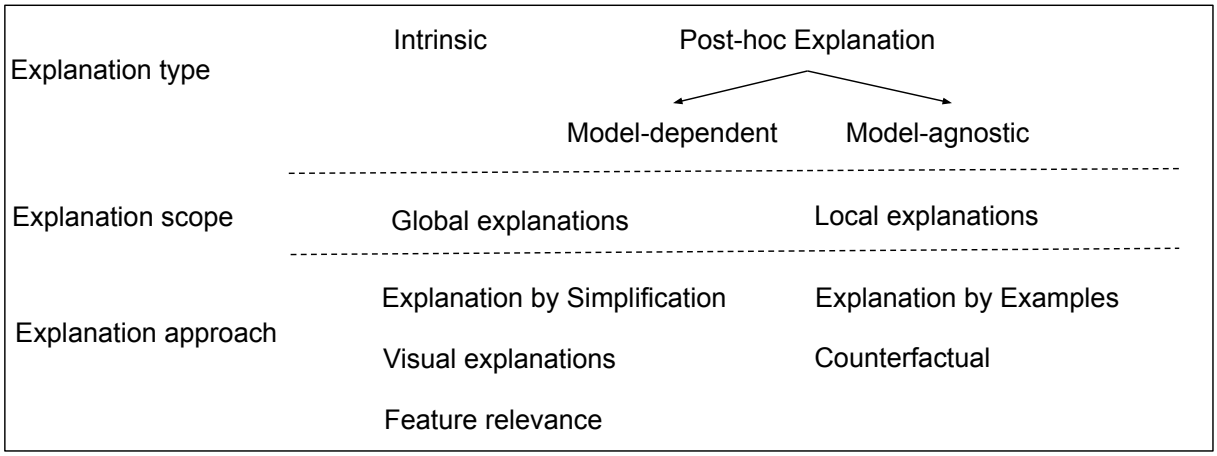
There are many questions we might want to be able to answer regarding regression models, such as: What does the generated prediction mean? What motivated a particular prediction? Is there any causal relationship? Can I trust the result? How can I improve the model? Several approaches have been proposed in the literature to answer these questions [88, 49, 18].

The field of *eXplanainable Artificial Intelligence* (xAI) has skyrocketed in the last few years, with hundreds of methods proposed to explain a black-box model. Several strategies were proposed, some of which are highly cited – such as LIME [106] and SHAP [76], and many recent works performed literature reviews of the field [79, 49, 16, 12, 3, 18]. There are also many attempts to provide useful measures to quantify the performance of explanatory methods [98, 83] and extensive benchmarks to evaluate which explanatory method has the best overall performance [9, 56, 119]. Some papers criticize the direction the field is going, pointing out pitfalls and challenges in explanation methods. [109, 40, 6].

2.1 Explaining regression models

Before going into details on how to explain complex regression models, let us first define the problem. Given a finite set of n input-output pairs representing the training instances $T = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ — with each pair $(\mathbf{x}, y) \in \mathbb{R}^d \times \mathbb{R}$ and $\mathbf{x}_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,d}\}$

Figure 2.1: Characteristics of methods for explaining regression problems. Methods differ in three aspects: the explanation type (whether it is intrinsic to the model or needs a *post-hoc* method), scope (if the whole model is explained at once or the model is explained for each prediction), and approach (there are different ways of providing explanations).



Source: created by the author.

— we define $\mathcal{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^T$ and $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$ as the input matrix $n \times d$, and the n-element output column vector, respectively. We then formulate the regression problem as learning a function $\widehat{f} : \mathbb{R}^d \to \mathbb{R}$ such that $\widehat{f}(\mathbf{x}_i) = y_i + \epsilon_i, \forall \mathbf{x}_i \in \mathcal{X}, \forall y_i \in \mathbf{y}$, where ϵ_i is the measurement error of the i-th data point.

Learning a regression model can be formalized as an optimization problem [16, 79], which we define as:

$$\widehat{f} = \arg\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \mathrm{L}(f(\mathbf{x}_i), y_i), \tag{2.1}$$

where $\widehat{f}$ is the best model found, $\mathcal{H}$ is the set of models (hypotheses) in the search space, and $\mathrm{L}(\cdot, \cdot)$ is a cost function to evaluate the error between the expected target value y_i and the predicted value generated by the model according to the values of the independent features $\mathbf{x}_i$[1].

The final model $\widehat{f}$, considered the best, can have different levels of complexity and make it possible – or not – for a human to understand the mechanisms that lead to the prediction of a certain output. This is when explanation methods become relevant.

At this point, it is important to mention that, following the definitions presented in [102, 62], we make a clear distinction between **intrinsically interpretable** models and **explanation methods** (those concerned with understanding the behavior of black-box models). Some authors use the terms interpretability and explainability interchangeably [88], but that is not the case here.

[1]The word *feature* is widely used in classification, whereas *variable* or *independent variable* is more comon in regression. We use the terms *feature* and *variable* interchangeably to refer to the different attributes of a dataset, being the latter preferred in the classical regression literature.

Fig. 2.1 illustrates three characteristics of explanation methods, starting from this distinction between intrinsically interpretable models and post-hoc explanation methods. Post-hoc explanation methods are built after the original black-box model and produce an auxiliary tool to help humans understand the outputs of the original black-box model. Concerning explanation methods, they can be model-dependent or model-agnostic, and can present explanations in different scopes – local or global.

Our primary interest here is in post-hoc explanation methods. Still, before going into detail on them, it is essential to say that even models intrinsically interpretable can have internal behaviors challenging for a human to understand. To give a few examples: consider complex mathematical expressions with many features, very deep decision trees or with many branches, extensive and complex rules, and a large number of instances considered relevant. Even simple structures become very difficult to interpret in the abovementioned cases and may require post-hoc explanations [53, 77].

Post-hoc explanation methods can be subdivided into two contexts: model-dependent and model-agnostic. Model-dependent explanation methods are tailored to the learning models they want to explain. Examples include explainability for Deep Neural Networks [38, 71, 113, 17], Support Vector Machines [48, 80, 59, 39], Bayesian networks [63], and Naive Bayes [90].

Model-agnostic methods, on the other hand, provide explanations regardless of the type of the original black-box model created [106, 36, 7, 48, 62, 102, 68, 51]. They include SHAP [76] and LIME [106], whose main approach is to locally replace a complex model with a linear model that is simpler to extract explanations from.

Regarding the scope of the explanation, methods can provide global or local explanations. While global explanations focus on understanding the general behavior of a model [88, 31], local explanations are usually interested in understanding the behavior of the model in a neighborhood close to an instance of specific interest [88, 31]. It is interesting to note that the method proposed in [102] denominates itself as intermediate, as its approach to understanding the black-box model works in a multilevel way, ranging from a global view to a local one.

Finally, there are many different approaches for generating post-hoc explanations, which can use visual explanation [3, 11, 36], feature importance [76, 106, 36, 21, 10], representative prototypes [51], and counterfactual explanation (an input that contradicts a prototype) [89] to reduce the lack of transparency of the model. Next, we first make a brief review of intrinsically interpretable models, followed by a more in-depth description of post-hoc explanation methods.

2.1.1 Intrinsically interpretable models

As previously stated, one of the goals of regression analysis is to understand a system of interest. In some situations, the interpretation of the model can be more important than the inference. Consequently, many methods were developed to find associations or the causal relation of the variables with the target feature. These methods generate inherently interpretable models (white-box) and provide a deeper understanding of the studied phenomena. In addition, some works that suggest that, in some scenarios, interpretable models should always be the choice [109, 55]. The main argument against explanation methods is that, since black-box models tend to model functions of high dimensionality and complexity, they may end up requiring explanation methods that are too complex to capture their behavior.

Following, we briefly review two types of white-box models and how to interpret their predictions, namely Linear Models and Tree-based Models.

2.1.1.1 Linear Models

A linear model describes the association between the d variables $\mathbf{x}$ and the output as a linear relationship of the form:

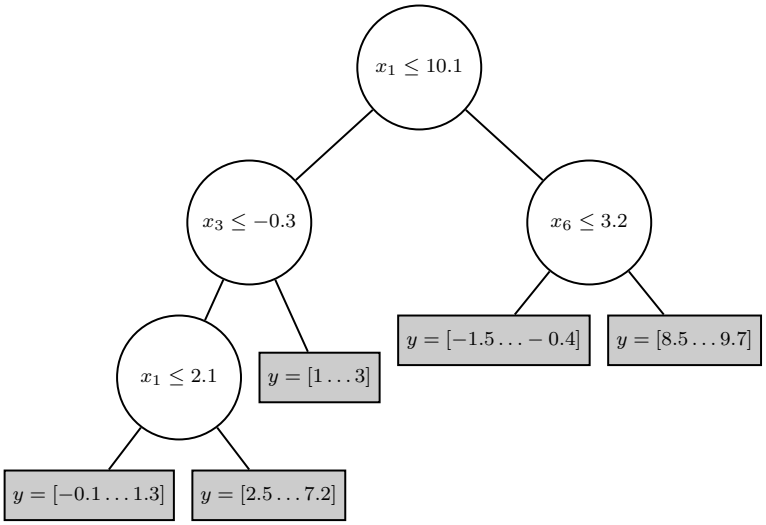
$$\widehat{f}(\mathbf{x}) = f(\mathbf{x}, \beta) = \beta_0 + \sum_{i=1}^{d} \beta_i x_i, \tag{2.2}$$

where β_0 is the intercept, β_i is the i-th coefficient of the independent variable x_i, and ϵ represents an unobserved random variable. This kind of regression model is parametric since the regression function is defined by a parameter vector β, and assumes the data has properties such as normality, homoscedasticity, and independence.

The coefficient of a linear model describes the association of the feature with the output, so when we observe a unitary change to x_i we expect to observe a change of β_i units in the output[2]. Knowing these associations can lead to a more thorough investigation of the studied systems. For the sake of the example, suppose we are minimizing the fuel consumption of a car, and our linear model says that a unitary change in the weight of the vehicle increases the consumption by 3 units. In that case, we can create car design simulations by replacing certain vehicle parts made of different materials and choosing the one with the most negligible weight while keeping the safety up to a threshold.

[2]Recall that association does not imply causality.

Figure 2.2: Example of a Decision Tree for regression. Inner nodes are predicates involving one input feature, a comparison operator, and a constant value. The leaf nodes contain information about the data instances corresponding to this path of the tree. This information can be a single prediction, an interval of values, or the distribution of values.



Source: created by the author.

Assuming that ϵ follows a normal distribution, we can calculate a confidence interval (CI) for each coefficient, which should be observed when looking at associations between features and the output. If the CI of a feature is strictly positive or negative, we have some insurance that the observed association will increase or decrease the output. On the other hand, if the interval contains a zero, it means we are uncertain of the effect of this feature, and, in some situations, we may choose to discard the features or follow up with a further investigation.

2.1.1.2 Tree-based Models

Tree-based models search for a binary tree structure describing a non-linear sequence of predicates that leads to a prediction. Each internal node of the tree is a predicate of the input features that returns either *True* or *False*. The leaves contain the prediction output for the target variable, being a numerical value for the regression task, or a categorical value for the classification task.

The main idea of a tree model is that each branch splits the decision space into two regions, maximizing the purity of the training data. For classification, purity means that the majority of the data instances in that region belong to the same class. For regression, it means that they differ by a small value ϵ.

Fig. 2.2 illustrates a decision tree for regression. The prediction starts at the root of

the tree, and, for every predicate that evaluates to *False*, it walks left. For those evaluated to *True*, it walks right. Whenever it reaches a leaf node, it applies an aggregation function to the values of the training instances following that path (*e.g.*, mean, median) to obtain the prediction.

Each path from the root to a leaf node can be seen as a conjunction of a list of predicates that are easy to understand. To illustrate, a prediction generated by going left then right in Fig. 2.2 means that $x_1 > 10.1$ and $x_3 \leq -0.3$. This gives us a discrete explanation of a prediction. Looking at this model, we can say that a large value of x_1 is associated with a prediction ranging between -0.1 and 7.2, and a smaller value of x_1 can either mean a prediction smaller within -1.5 and -0.4 prediction if x_6 is large or a high-valuated prediction if x_6 is smaller.

From this explanation, we can see that this type of model is capable of capturing the interaction of the features and the non-linearity of the generating process. We can also calculate the feature importance of this model as the sum of the importance of nodes involving that feature. The importance of a node is the product of how much it improves the optimization criteria by the probability of reaching that node.

Despite the limitations of this model imposed by how space divisions are made, some tree-based models are among the most accurate when applied together with bagging (*e.g.*, Random Forest) and boosting (*e.g.*, Gradient Tree Boosting) techniques. However, tree ensembles have reduced interpretability when compared to stand-alone tree models.

2.1.2 Gray-box models

Gray-box models are in-between the white-box and black-box spectrum, being partially interpretable. A gray-box model is one where symbols are partially interpretable [118], or uses a mixture of black-box and white-box models [75], or makes it possible to partially dazzle their internal design, structure and implementation [3].

One example of gray-models in the context of regression analysis is Symbolic Regression (SR) [95, 70]. This procedure returns a mathematical expression that can be further analyzed to obtain insights into the prediction process.

SR has the potential to find the governing equation of a dataset if the equation can be expressed through the mathematical elements used by SR in the search procedure. However, SR is often seen as a challenging task and demands a larger sample set than traditional regression techniques [60] since it is non-parametric and optimizes both the expression and its free-parameters together.

As SR returns an analytic model, it is possible to apply analysis related to the

effects and associations of a feature similar to what we have with linear regression. For instance, we can study the association of a variable and the output with the partial derivatives of the model. The partial derivatives return a function describing the observable change of the output when slightly changing a certain feature. In linear models, this change is constant, but, in non-linear models, it can vary depending on the region of interest.

2.1.3 Post-hoc explanation methods

In the scope of post-hoc methods of explanation we have several types of explanation recurrently found in the literature, including those reported in Fig. 2.1: *i)* explanation by simplification; *ii)* visual explanations; *iii)* feature relevance; *iv)* explanation by examples; and *v)* counterfactual.

Approaches based on *explanation by simplification* modify the structure of the resulting model to make it easier to understand. Simplifications can be elaborated in different ways. One possibility is to perform mathematical substitutions generating equivalent algebraic expressions, generally shorter and with more trivial operands to understand. The model must be constructed by composing a mathematical expression. Another possibility is to replace the original model with substitute models that try to approximate the original and more complex model in a given region of the solution space. For instance, a white-box model, such as linear regression, can be used to generate a local explanation for specific outputs of the original model. This second approach is used internally in various explanation methods such as LIME (Local Interpretable Model-Agnostic Explanations) [106]. GPX (Genetic Programming Explainer) [35] and Symbolic Meta-modeling [7] use an evolutionary algorithm to produce a mathematical expression capable of explaining the region of the sample.

Concerning *visual explanations*, we find methods that use graphical representations to show how the different features that describe the problems can interact or even contribute to modifying the predicted target output. Examples of visual explanations include the Individual Conditional Expectation (ICE) [3] and Partial Dependence Plot (PDP) [121] explanation methods, which provide a global visual explanation of the model of interest. ICE shows the dependence between a specific input feature and the output, while PDP considers the dependence of the effect resource average and the output. Additionally to PDP, Accumulated Local Effects (ALE) plots explains visually the model, which is more appropriate to understand if there is a correlation between the different features of the model. For that, ALE calculates the difference between the outputs along

a reduced interval for a feature of interest.

Other visual explanation methods proposed in the literature are dedicated to specifically elucidating models that work with images, *e.g.*, LIMEcraft [57] and XRAI [66], and work by identifying the specific regions of the image that most influence a given model prediction.

Explanations based on feature relevance, in turn, are probably the most explored type of explanations explored in the literature with several representatives, among them: SHAP (SHapley Additive exPlanations) [76], LIME, GPX, SAGE (Shapley Additive Global importancE) [21], Partial Effects [10], and Integrated Gradients [114].

These methods follow three main approaches. The first tries to approximate the black-box with an interpretable model and then uses the interpretable model as a proxy to explain the black-box model. The second removes the features from the model, then measures how much the behavior of the model changes in the absence of the removed features. In practice, to remove a feature, one would need to modify the inner structure of the black-box model. As this is not possible, these methods fix it to specific values or permute the feature to make it uncorrelated, behaving as a noise feature. The third approach uses the information of the gradient or sensitivity to small changes on the input — calculated by approximations — to determine how the model prediction varies with input variations.

Concerning methods following the first approach, which generates local approximations of the black-box model, LIME is a pioneer and probably one of the most popular. It explains an observation locally by generating a linear model (considered human-interpretable) using random instances generated in the neighborhood of the instance of interest. The linear model is then used to portray the importance of features based on the coefficients of the model.

Extrapolating the limit of explaining the regression models only by means of linear functions, [7] introduce the Symbolic Metamodeling approach. In this approach, in order to create a complete mathematical description of the model, a replacement for the black-box is created using a symbolic regression with familiar mathematical functions combined by elementary arithmetic operations such as addition and multiplication. For datasets with a large number of features, this method may be infeasible, as it would make it considerably harder to understand the equations provided and the possible interactions between the features. In addition, the space of potentially used mathematical functions must be carefully evaluated for each type of user to avoid losing the desired explanatory power and simply replacing a black-box model with another.

Considering that the synthetic generation of neighbors is an important step in several methods of explanation, [62] proposes a new approach based on the LID measure (Local Intrinsic Dimensionality) capable of measuring the quality of the generated neighbors. The idea behind LID is to represent the distribution of data from the vicinity of

an instance of interest and, therefore, a neighborhood with an average LID close to the LID of a test instance would be ideal. This method was named LEAP (Local embedding aided perturbation), and it works by initially adjusting the local subspace around a given instance using the LID of the test instance as the destination dimensionality, and then generating neighbors at the local embedding and projecting them back into the original space. The proposed technique can be used as a modular procedure in any existing local explanation method and in experiments performed it has shown to be able to generate more realistic neighborhoods than the methods previously described in the literature. To estimate the LID value, the work uses the maximum likelihood estimate and the Euclidean distance. As we know, especially for high dimensional data, the Euclidean distance can lose significance. In addition, the work does not use the value of the model output to generate the neighborhood, instead, the neighborhood ends up being based on the initial training dataset and not on the black box model that is the target of explainability.

[122] argues that the most important aspects of perturbation-based local explanation methods for finding the nearest neighbors of an instance of interest are stability and reproducibility. They consider that repeated executions of the algorithm under the same conditions should ideally produce the same results. Therefore, a hypothesis testing framework based on the central limit theorem, called S-LIME, is presented to automatically determine the number of disturbance samples that are needed to guarantee the stability of the resulting explanation.

SHAP is another popular method following the second category (feature removal) of feature importance methods. It is based on a concept from coalition game theory: the importance of a feature d is given by evaluating, for each possible subset of features in the problem, the changes in the model's performance when d is present in or absent from a given subset [76]. Each subset is considered because the impact of a feature may depend on its co-features. Since a feature cannot be removed, its absence is simulated by setting it to a reference value.

The method SAGE also uses game theory and the Shapley value to understand interactions between features, showing how each feature contributes to the model's observed loss. As argued by SAGE authors in [21], SHAP global explanations were applied only heuristically, observing output changes instead of loss decrease.

Unlike the methods previously presented, [101] proposes a multilevel explanation based on a decision tree set (explaining different degrees of cohesion depending on the level of the tree) aggregated with independent local explanability on the leaves (similar to LIME). The approach, called MAME (Model Agnostic Multilevel Explanation), provides both global, intermediate and local explanations. As criticisms, we have once again the use of only Euclidean distance in the generation of the explanations. Additionally, the quantitative testing was limited to only one regression dataset, which contained 392 instances and 7 attributes. In addition, for the specific case of regression, the obtained

result was very similar to that of the main baseline, Two Step (which is a hierarchical convex clustering [19, 117] where a median explanation is computed for each cluster), and outperformed SP-LIME. Observations like these make the work carried out less conclusive for use in explaining regression problems.

Partial Effects, in turn, is an example of the third approach followed by feature importance explanation methods and explores the partial derivatives of the expected values for the features concerning the regressor. It quantifies the amount of change that a modification in a feature value would cause on the output, if the other features were fixed at a representative value. It relates to PDP since the Partial Effects represent the derivatives of the Partial Dependences.

In this same category, we have the Integrated Gradients method, which calculates the gradient of the model's prediction output to its input features and thereby computes its importance.

In the field of *explanations by examples* (or prototypes), relevant instances are crafted or chosen from training data to explain the whole model or local predictions. This type of explanation is similar to the human way of interpreting problems by analogy. As pointed out in the work of [88], any clustering algorithm that returns instance points as cluster centers would qualify to select prototypes. Similarly, prototypes could be found using hub points. Hub points are training instances that appear unusually often among the k nearest neighbors of other instances [74]. Hub points could be used as prototypes since the frequency in the list of neighbors of many points is an indicative of similarity with many examples, which is a desirable property for prototypes. In the case of instance selection for classification problems, a common approach is to maintain/identify the class edge instances [94, 111]. There are also few works on instances selection for regression problems [111, 69, 126, 86]. A method that manages to obtain good results in this task and that we can easily adapted to our problem is the traditional single objective Genetic Algorithm (GA), where each individual will be composed of k prototypes and the evolutionary process will take place in order to minimize the error obtained.

In work [105] SP-LIME (Submodular Pick - LIME) was presented, a method that provides a global explanation of the evaluated model. Therefore, the goal is to make a non-redundant selection of instances in relation to the features of the problem. As negative points of this method are the limitation of taking into account only the diverse features to select the explanatory instances, disregarding characteristics such as the capacity of the instances to reconstruct the model, common approach to analyze works whose purpose is to select instances. The presented baseline can also be considered fragile since the implementation is compared only with a random selection of instances, known approaches in other areas such as clustering could be compared to the presented proposal.

Among methods for instance selection in regression, the authors in [14] propose a method based on the set coverage problem, which has as a significant advantage the

automatic choice of the appropriate number of prototypes. [68] propose MMD-critic, a method based on Maximum Mean Discrepancy (MMD), a measure that captures data distribution. Based on this metric, MMD-critic selects prototypes and criticism (*e.g.*, outliers) and compares the training data distribution and the distribution of the candidate prototypes by MMD. The method selects the set of prototypes that minimize the discrepancy between these two distributions.

MMD-critic is extended in [51], generating ProtoDash. ProtoDash returns, together with the generated prototypes, a set of non-negative weights indicating their importance. It is not directly intuitive for ProtoDash to explain the regression model, but the data. Thus, the authors show that both in problems of explanability in regression and in classification ProtoDash is capable of achieving results equal to or better than other related works in the literature. Despite explicitly dealing with a regression problem, we see some limitations in this work: i) a single database in the context of a regression problem was evaluated: 80 million customers of a large retailer whose objective is to predict total spending per customer; ii) only RMSE was considered to quantitatively evaluate the method compared to competitors; iii) in the quantitative study by the RMSE metrics, a set of instances was selected that would hardly help a human to understand the problem: 1.5 million (after this value, the improvement in the objective became incremental). In the qualitative study, experts were shown a much smaller subset, 100 instances (a value that we still consider high to be analyzed by a human being), which reinforces our intuition that the metrics RMSE is not the most adequate to evaluate explanations; and iv) no more formal studies were carried out capable of guaranteeing the statistical difference between the proposed method and the analyzed competitors.

In another context, [52] proposes a model that uses hierarchically organized prototypes to classify images, making it possible to find different explanations for a prediction at each level of the taxonomy. [85], in turn, use prototypes allied to Recurrent Neural Networks (RNNs) to provide interpretability for classification in sequential data. They highlight the importance of prototypes being simple, sparse, and diverse.

In *counterfactual explanations* the objective is to find an input that contradicts an instance of interest and use this instance to circumvent the lack of transparency of the model. An example of a method in this category is DiCE (Diverse Counterfactual Explanations) [89]. This type of explanation can be of great value in many contexts, *i.e.*, when we want to identify unfair models or those that considerably modify the output when looking at some sensitive characteristic that should be protected, such as gender or race [45].

Another method of this approach is LORE (LOcal Rule-based Explanations), introduced in [48], provides explanations based on decision and counterfactual rules, allowing the user to identify the minimum modification required to assign a different class. Although originally designed for non-regression problems, we believe LORE could be easily

adapted. The method uses a genetic algorithm to choose the local neighborhood based on similarities between instances in relation to the features and the desired class (for two-class problems). To extend the method to regression problems, neighbors could be selected based on not only feature distance but also model output for each instance, and the binary decision tree could be replaced with an appropriate explainer for regression.

Finally, several other works dedicated exclusively to classification problems could be used for regression if we can map the output classes to successive integers, regardless of the loss of precision resulting by this choice. Some works in this line are: i) Anchors [108]: find anchor rules, which are usually easy to interpret and capture non-linear relationships. They are sufficient rules where (almost always) changes in other non-present resource values do not affect the predict; ii) LACE (Local Agnostic attribute Contribution Explanation) [97]: method also based on rules that assesses the effect of attributes and values, either individually or together, on the output assigned to a given test instance. Such rules are learned locally in the neighbors of the test instance, with the use of an associative classifier, and the relevant characteristics found for the classification are evaluated in subsets without the need to use brute force on all initially possible combinations, which would be unfeasible in real applications. The neighborhood is considered as the k training elements closest to the test instance (a specific method to determine the k value has not been proposed); and iii) CEC (Constraints based Explanation for Classifications) [27]: looking for the least semantically valid modification to change the class, for this purpose this method requires access to the complete model of the classifier and clear definitions of the restrictions of the problem domain in order to carry out local perturbations.

Table 2.1 presents a summary of the methods discussed in this section.

2.2 Evaluation of explanation methods

Evaluating explanations is not a simple task. [31] discusses how the explanations can be evaluated. Three possibilities are listed:

1. Application level evaluation (real task): domain experts participate in the evaluation;

2. Human level evaluation (simple task): laymen are used to test more general functions of the quality of an explanation; and

3. Function level evaluation (proxy task): it does not require humans and uses previous knowledge. In this case, it is assumed that someone has tested it with humans

Table 2.1: Summary of explanation methods for regression. Methods indicated with an * have their explanations evaluated in the in the Appendix A of this work.

Method	**Type**	**Scope**	**Approach**
Linear model (*)	intrinsic	global	feature relevance
Decision tree (*)	intrinsic	global	visual
SR GP-NLS (*)	intrinsic/post-hoc	global	feature relevance
SHAP (*)	post-hoc	local	feature relevance
LIME (*)	post-hoc	local	feature relevance
SAGE (*)	post-hoc	global	feature relevance
Partial Effects (*)	post-hoc	local/global	feature relevance
Integrated Gradients	post-hoc	local	feature relevance
MAME	post-hoc	intermediary	feature relevance
GPX	post-hoc	local	feature relevance and visual
Symbolic Metamodeling	post-hoc	local/global	feature relevance and visual
ICE/PDP (*)	post-hoc	global	visual
ALE	post-hoc	global	visual
SP-LIME	post-hoc	global	explanation by examples
MMD-critic	post-hoc	global	explanation by examples
ProtoDash (*)	post-hoc	global	explanation by examples
DiCE (*)	post-hoc	local	counterfactual
LORE	post-hoc	local	counterfactual

before. For example, it is known that humans can interpret the simple results of linear regression or decision rules, so this knowledge is used during evaluation.

In the specific case of works that deal with explanations of regression problems, we can observe that there is no agreement on how to evaluate the quality of the results obtained, as shown in Table 2.2. Domain experts contributed to the evaluation for MAME and ProtoDash. This is expected, as this type of evaluation requires a greater amount of time and resources. For MAME, the evaluation involved experts from the oil and gas industry, as they used a real world dataset regarding industrial pump failure. In Protodash, two datasets were used in this evaluation context. The first refers to retail electronic commerce, and the task was to predict customer spending and possible loyalty. The second dataset refers to a set of questionnaires produced by the US Department of Health.

In general, lay users contributed to the evaluations of the LIME, SP-LIME, ProSeNet and MAME methods. In LIME and SP-LIME, users were used to select the best explanation model presented after the insertion of noise in the datasets. In addition, LIME also assessed whether users can gain new insights into the model when faced with the explanation provided and in SP-LIME assessed the user's ability to improve the model, for example, indicating which features could be removed from the set of training. In ProseNet, the Amazon Mechanical Turk was used to assess how comprehensive and accurate the explanations of the input strings were in a sentiment classification dataset. MAME

Table 2.2: Mechanisms used to evaluate the quality of results obtained by regression models.

Reference	Evaluation metrics
	Feature-based explanations
LIME [107]	Average of recovered features compared with interpretable algorithms Average F1 of trustworthiness Adding Noisy Data Users on Amazon Mechanical Turk
SP-LIME [107]	Non-specialist humans
Symb. Met. [7]	R2 Median of selected features Verification of most important features in a real dataset
LORE [48]	Average accuracy of rules and coverage Jaccard coefficient Average hit rate Predict fidelity Comparison of density and location of the neighborhood Qualitative study
LEAP [62]	Cosine similarity F1
S-LIME [122]	Jaccard index
MAME [101]	Generalized Fidelity metric Feature importance rank correlation Non-specialist humans Domain experts
	Examples-based explanations
PS [14]	Classification error
MMD-critic [68]	Classification error
ProtoDash [51]	Accuracy RMSE Domain experts
HPnet [52]	Accuracy
ProSeNet [85]	Accuracy User on Amazon Mechanical Turk

was also evaluated by non-specialists using a database in the context of public loans, where evaluators decided on the approval or rejection of a person's loan application based on explanations about their financial characteristics.

Finally, other types of evaluations were diversified and mostly based on quantitative analysis metrics. These included the accuracy or error of the explanatory model (PS, MMD-critic, ProtoDash, HPnet, ProSeNet, and LORE), the RMSE when the explanations

were used to reconstruct a given model (ProtoDash), the cosine similarity obtained in the explanation (LEAP), the R2 of the function discovered with the true function in synthetic bases (Symbolic Metamodeling), the fraction of features that were recovered in the explanation when compared with inherently interpretable algorithms (LIME), and the Jaccard index to measure the similarity of the top 5 selected features in different executions to explain a specific instance (S-LIME) to measure the stability of explanations.

2.3 Contracting the output obtained by different explanation methods

In Appendix A, we selected some of the most popular methods to contrasts their results in terms of prediction error and explanations. For that, we will use one synthetic and one real-world dataset.

In this way, we show that often white-box regression models (*e.g.,* linear or decision trees) or gray-box models (considered partially interpretable) can provide an appropriate intrinsic interpretation to the model. By understanding and evaluating different explanation methods, we point out which methods are complementary and those that superpose in terms of explanations. Based on that, we indicate a set of methods that, when used together, can give the user the best possible explanation using different types of model understanding.

Concerning methods that use features to provide explanations, our discussion shows that SAGE (Shapley Additive Global importancE) [21] can show the importance of each feature (around an average), and Partial Dependence Plots (PDP) [121] explains the behavior of the model by varying the values of the features. Concerning changes in instances, ProtoDash [51], in turn, shows which are the most representative instances of the training set and uses them to explain new methods. DiCE (Diverse Counterfactual Explanations) [89] shows how minimal changes in specific instances are capable of significantly altering the output for a given value. On the other hand, we also observed methods that provide similar explanations, whose concomitant use will not bring much new information to the user, e.g., PDP together with ALE (Accumulated Local Effects) [11].

2.4 Summary

This chapter we illustrated and explored different explanation techniques for regression models, from inherently interpretable models to model-agnostic explainers.

Regarding the interpretable models, linear regression provides a simple and clear explanation for the chosen datasets. Even though it is easy to understand the behavior of the model, it does not reflect the reality of the data. The same can be said about the decision trees, which even though can handle non-linearity, the discretization of the model may create a rough explanation of the system's behavior.

With Symbolic Regression, we have an analytical model that the practitioner can manually inspect. Since it returns a non-linear model, it requires some support tools to fully understand its behavior. Combined with Partial Effects, this model can provide a pointwise association analysis of every variable to the target.

When we have a black box model we need explanation tools to extract useful information from the generated model. A similar interpretation to the linear coefficients and partial effect can be achieved using graphical explanations such as Partial Dependence and Accumulated Local Effects plots.

Another form of explanation is those that revolve around a single data point, called local explanation. In this situation, we can measure the local variance of a variable using LIME, with an interpretation similar to a linear model. We can also measure the contribution of each variable to the prediction when compared to a baseline data point (usually the average prediction) with SHAP. Another possible global explanation is to find representative points with ProtoDash, such that they encompass a different region of the prediction space. Those points can be studied individually to understand the behavior of the model in each defined region.

Finally, we can search for the slight change in the data point that makes a change in the prediction such that it reaches the desired target. This is called counterfactual explanation, represented by DiCE in this work, and it can give an insight of the possible changes we can make to a single point to reach a desired goal.

As we can see from all these possible explanations, we can extract different information from our models that can help us gather useful information from the system we are studying or the behavior of the model itself. It is important that, when analyzing the prediction model, the practitioner takes time to understand the output of each explanation method to go beyond the feature importance analysis. By doing so, they can find important questions and answers about the studied dataset.

Chapter 3

ELA: A feature-based explanation method for regression

In this chapter we present our first contribution for explanation methods based on features. The proposed approach, named Explanation by Local Approximation (ELA), is simple, effective and model agnostic (i.e., it can be applied to any regression model): it finds the nearest neighbors of the point we want to explain and performs a linear regression using this subset of points. This approach uses the coefficients of the linear regression to generate a local explanation to the model. We used a linear regression in this process because it is a well-known and inherently interpretable method. Moreover, this algorithm allows for a clear understanding of how each feature contributes to the local output of the model, making it an ideal choice for our purposes.

It is important to note that while the ELA method is model-agnostic, it is desirable to have continuous-valued features for the problem being analyzed. This is because local explanations are obtained using linear models, which require continuous features to generate meaningful explanations. If categorical or discrete features are present, they can be transformed into continuous features through pre-processing techniques. However, this transformation should be done carefully, as it may affect the performance and accuracy of the model. Therefore, when possible, it is preferable to use continuous features in order to obtain more reliable and accurate local explanations.

The experiments carried out in this step used as a regression model to be explained, constructed by a Genetic Programming (GP) algorithm. GP is a bio-inspired method, where a population of solutions representing models is evolved for a number of generations. The models are evaluated according to a fitness function, and probabilistic selected to undergo mutation and crossover operators. One of the main advantages of models evolved by GPs is the fact that the evolved models can be interpreted by humans [32, 81]. However, in the same way as other supervised models that are considered interpretable – including decision trees, decision rules or linear regression – the more complex the model becomes, the less interpretable it is.

It will be shown in our experimental study, when non-linear relationships are present in the data, even for synthetic datasets with two attributes (explainable vari-

ables), as the maximum depth of the GP tree increases (potentially increasing the model complexity) and the root mean square error (RMSE) decreases, the number of nodes in the tree grows from an average of 5 nodes (maximum depth 2) to more than 65 nodes (maximum depth 6), making the original model difficult for human understanding

We perform a quantitative analysis of the proposed approach in a set of 10 synthetic benchmarks with known non-linear relations between the predictive and target variables. We then present a qualitative analysis in a real-world dataset where we do not need very specialized knowledge to interpret the resulting justifications, as evaluation of the interpretability of the models is complicated and, for most datasets, require specialized expertise.

Results show that the errors of the local approximations are similar to those of the regression performed with all points. It also shows that simple visualizations can provide insights to the users about the most relevant model attributes. Finally, we also concatenate the results of many local interpretations into a single, global explanation, which can also aid the process of understanding predictions.

After looking at these initial results, Chapter 4 presents DELA (Dynamic Explanation by Local Approximation). In this case, we will study which distance measures are most appropriate for each dataset. In addition, we will perform experiments considering a variable neighborhood size instead of a fixed neighborhood. Finally, we will also improve the calculation of the local feature importance.

The remainder of this chapter is organized as follows. Section 3.1 introduces the proposed method, while Section 3.2 presents the quantitative and qualitative experiments. Finally, Section 3.3 draws conclusions.

3.1 Method outline

This section describes ELA (Explanation by Local Approximation) in the context of symbolic regression with GP, although ELA can be easily generalized to any other type of regression method.

Let us assume we have a training set $T = p_i = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ and a test set $T' = p_j = \{(\mathbf{x}_j)\}_{j=1}^{m}$ — with $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$. ELA works by generating a non-linear regression model (e.g., symbolic regression) for predicting the value y' of data points in T', and then performs a local approximation with a linear regression method to generate explanations for the predictions generated by the first model, as illustrated in Algorithm 1.

The algorithm receives as input the training and a test sets, T and T', and the

Algorithm 1 Explanation by Local Approximation (ELA)

Require: T (train), T' (test), k (# of neighbors)
1: SR = Run GP(T)
2: **for** each p $\in T'$ **do**
3: y'= SR(p)
4: NN = Find k nearest neighbours of p considering only attributes in SR
5: LR = Run linear regression(NN)
6: Compute importance of attributes in LR
7: Show local explanation for predicting y'
8: **end for**
9: Plot global explanation

value k of the number of neighbours to be considered in the local explanation. First, we run the GP in the training set to find the function that describes in the T (line 1). Having this function, we explain (or justify) its prediction for each test point considering a local linear regression over a set of k neighbour points. Note that at this point we could use any other regression algorithm that creates a model capable of making output predictions on the instances.

Given a test point p, we first use the original GP model SR to find the predicted value y' (line 3). Each GP function is represented by a tree, generated respecting a maximum tree depth. The GP functions are evaluated using the RMSE as the fitness function. A tournament selection is used to select the individuals that undergo crossover and mutation operators.

Next, we find the k neighbours of p (line 4 and Equation 3.1). We use a Euclidean distance in this process, as shown in Equation 3.2.

$$NN_p = \arg\min_{p_i}\{dist(x, x_i)\} \tag{3.1}$$

$$dist(x, x_i) = \sqrt{\sum_{a=1}^{d} w_a (x_a - x_{ia})^2} \tag{3.2}$$

In this equation, d is the number of predictive attributes, and w_a represents the weight of the a-th attribute selected by the GP. If the attribute is present in the SR function returned by the GP, its weight is set to 1, and the attribute is considered in the calculation. Otherwise, the weight is set to 0.

Having the k nearest points of an examples p, we use a linear regression method to find the function that best represents these k points (line 5). This linear equation is able to provide a local explanation of the prediction given to p considering its neighbours. Being a linear equation, the interpretability of LR is more straightforward than the interpretability of the function returned by the GP. Of course there are exceptions, specially if the number

of nodes of the GP tree is small. However, note that this method is recommended for cases where the original function SR is difficult to be read and understood by a human.

Next, we analyze the coefficients of LR by calculating a measure of importance of each attribute x to the final prediction (line 6). Importance is defined as the contribution that each attribute exerts to the final value of the output of p, given by y'. As shown in Equation 3.3, we look at the module of the importance of each attribute by multiplying its coefficient by its value in p and normalizing it.

$$Importance_{x_i} = \frac{|coefficient_{x_i} \times x_i| \times 100}{\sum_{a=1}^{d} |coefficient_{x_a} \times x_a|} \tag{3.3}$$

This measure of importance, where the signal is ignored, is able to provide extra information about the proportional contribution of each attribute during the regression task, both from a local or a global. Recall that the user can also access the coefficients found by the local linear regression directly, providing an additional way of explaining the result obtained by the regressor.

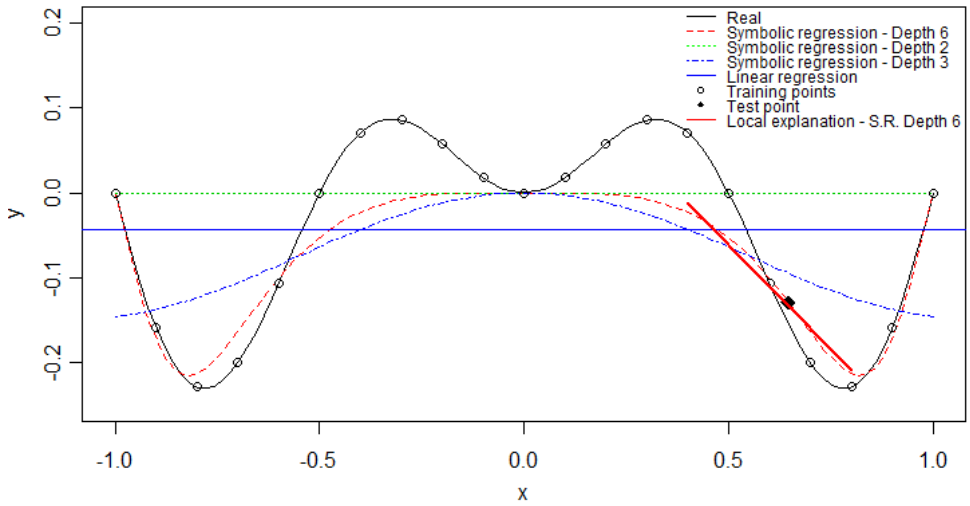
In order to provide a visual interpretation of the result to the user, we use the following procedure (line 7). We start by checking how different are the outputs of the K training examples in NN from the test point of interest p. As they are neighbours, we expect than to be similar. We consider that a difference higher than 10% of the maximum range of the output values in T should be disregarded. For example, if the outputs y in T are in the interval [1,10], a variation of 1 in the neighborhood is considered as the threshold of similarity for visual explanations. This threshold is used to reduce the risk of considering noisy data in the explanations, damaging the method. For the subset of neighbours in NN within this defined threshold, we look at the variation of the values of each attribute x in the local explanation. The process aforementioned is repeated for each test set, providing a justification for each prediction.

Finally, the proposed method also presents an global explanation of the behavior of the symbolic regression obtained when we have a large set of explanations for different test points. In order to obtain that, all test points are discretized according to the output variable in z equally spaced intervals. For each interval, the average importance of the subset of input attributes x is calculated. The result is then plotted on a graph of stacked areas, given insights of the importance of different attributes to the whole dataset (see Figure 3.3).

Figure 3.1 illustrates the results of ELA using one of the synthetic datasets considered in the experimental analyzes reported in Section 3.2. The real function, shown in black in the figure, is defined by Equation 3.4.

$$f(x) = 0.3x \sin(2\pi x) \tag{3.4}$$

Figure 3.1: Example of the approximation performed by the ELA method (red line) for the synthetic dataset keijzer-1 (black line).



Source: created by the author.

The test point p we want an explanation for is represented by the black diamond, and the red line going over the black diamond was generated by ELA. The other functions shown in the figure were generated by the GP using different maximum depths (a parameter that has a direct impact on model interpretability) and a simple linear regression using the whole set of training points in T. Note that the line produced by ELA follows the same direction of the function generated by the symbolic regression with depth 6, also represented in red.

3.2 Experimental Analysis

This section presents an experimental analysis of the interpretability of the proposed method. The symbolic regression method was implemented using the Python package Deap[37]. The method was also implemented in Python and is available for download[1].

The results of the proposed method are compared to the original function found by the GP and to a linear regression method run with no regularization, an L1 and an L2 normalization [15]. These models were chosen as they are considered interpretable regression methods, as the coefficients give us a notion of attribute importance.

Results are analyzed in three steps. First we perform an analysis of the different methods using all training points, considering both RMSE and model complexity. Next, we make a quantitative analysis of ELA using 10 synthetic datasets. Finally, we perform

[1]https://github.com/renatomir/ELA-WCCI2020

Table 3.1: Dataset used in the experiments of the explanatory ELA method.

Dataset	**# Attributes**	**# Train**	**# Test**	**Nature**
keijzer-1	2	21	2001	Synthetic
keijzer-4	2	101	101	Synthetic
keijzer-7	2	100	991	Synthetic
vladislavleva-1	3	100	2025	Synthetic
vladislavleva-2	2	100	221	Synthetic
vladislavleva-3	3	600	5083	Synthetic
vladislavleva-4	6	1024	5000	Synthetic
vladislavleva-5	4	300	2700	Synthetic
vladislavleva-7	3	300	1000	Synthetic
vladislavleva-8	3	50	1089	Synthetic
wineRed	12	1279	320	Real

a qualitative case study considering a real-world dataset, namely the *wineRed* dataset.

3.2.1 Experimental Setup

We tested the proposed method in a set of 10 synthetic and one real-world dataset, which will be later used in a qualitative case study. The datasets are described in Table 3.1, which shows the total number of attributes (predictive and target (Attributes), number of data points used in the training (Train) and test (Test) phases. All synthetic datasets were originally presented in [67]. The real dataset is available at the UCI repository [28, 20]. Note that we varied the size of the training set in order to observe its impact on the model's explanation.

The GP functions are defined by the binary operations of addition (+), subtraction (-) and multiplication (x), in addition to the analytic quotient (AQ) [92], which has the general properties of division but without discontinuity (see Equation 3.5). The terminals are the predictive attributes of the datasets.

$$Aq(m, n) = \frac{m}{\sqrt{1 + n^2}} \tag{3.5}$$

After a preliminary parameter tuning, the GP was executed with an initial population of 1,000 individuals evolved for 250 generations, using a tournament selection of size 7. The probabilities of crossover and mutation were defined as 0.8 and 0.2, respectively. The depth limit of the trees was varied to show the differences in the number of nodes of the solutions found and their impact in interpretability.

Table 3.2: Mean and standard deviation of the RMSE obtained by regression models on the training set.

Dataset	Genetic Programming			Linear Regression		
	Depth 2	Depth 3	Depth 6	LR	L1	L2
keijzer-1	0.120 (0.000)	0.090 (0.002)	**0.041 (0.012)**	0.110	0.110	0.110
Keijzer-4	0.320 (0.000)	0.320 (0.000)	**0.272 (0.057)**	0.320	0.320	0.320
Keijzer-7	0.990 (0.000)	0.515 (0.075)	**0.150 (0.061)**	0.410	0.410	0.410
Vladislavleva-1	0.131 (0.005)	0.103 (0.010)	**0.055 (0.014)**	0.130	0.230	0.130
Vladislavleva-2	0.320 (0.000)	0.319 (0.004)	**0.228 (0.057)**	0.320	0.320	0.320
Vladislavleva-3	1.090 (0.000)	1.087 (0.005)	**0.913 (0.132)**	1.090	1.090	1.090
Vladislavleva-4	0.193 (0.005)	0.181 (0.003)	**0.158 (0.011)**	0.190	0.190	0.190
Vladislavleva-5	0.346 (0.034)	0.290 (0.069)	**0.118 (0.075)**	0.600	0.610	0.600
Vladislavleva-7	3.372 (0.011)	2.648 (0.385)	**1.738 (0.239)**	3.670	3.680	3.670
Vladislavleva-8	1.719 (0.002)	1.490 (0.056)	**0.764 (0.108)**	1.760	1.780	1.760

Table 3.3: Mean and standard deviation of the RMSE obtained by regression models on the test set.

Dataset	Genetic Programming			Linear Regression		
	Depth 2	Depth 3	Depth 6	LR	L1	L2
keijzer-1	0.120 (0.000)	0.080 (0.002)	**0.042 (0.012)**	0.110	0.110	0.110
Keijzer-4	0.320 (0.000)	0.320 (0.000)	**0.272 (0.057)**	0.320	0.320	0.320
Keijzer-7	0.960 (0.000)	0.510 (0.074)	**0.149 (0.060)**	0.380	0.380	0.380
Vladislavleva-1	0.151 (0.004)	0.127 (0.008)	**0.096 (0.025)**	0.190	0.210	0.190
Vladislavleva-2	0.300 (0.000)	0.300 (0.003)	**0.217 (0.053)**	0.300	0.300	0.300
Vladislavleva-3	1.010 (0.000)	1.006 (0.005)	**0.855 (0.123)**	1.000	1.010	1.000
Vladislavleva-4	0.209 (0.004)	0.196 (0.008)	**0.166 (0.014)**	0.190	0.190	0.190
Vladislavleva-5	0.508 (0.043)	0.443 (0.088)	**0.214 (0.113)**	0.840	0.840	0.840
Vladislavleva-7	3.763 (0.014)	3.031 (0.423)	**2.070 (0.288)**	4.050	4.040	4.050
Vladislavleva-8	2.225 (0.020)	2.166 (0.085)	**1.537 (0.346)**	2.280	2.280	2.280

Considering the non-deterministic character of the results obtained by GP, all tests presented were executed 30 times, and the average RMSE and number of tree nodes are reported.

3.2.2 Regression with all points

We first analyze the error results of the symbolic regression and linear regression methods when run with all test points. This is important for two reasons: first, to show we need more than a linear regression to solve the problem; second, to ensure the results of the local explanations will not increase the test error.

Tables 3.2 and 3.3 shows the mean RMSE followed by their standard deviations in the training and test sets, respectively. Three maximum depths of GP trees were considered: 2, 3 and 6, to show the impact of the complexity of the model to the RMSE. The linear regression was run without any regularization and using both L1 and L2. The parameter that defines the regularization force (alpha) of the methods with L1 and L2 was set at 1.0.

Observe that the results of RMSE of the GP improved as we increased the depth of the tree. The results of the linear regression with and without regularization present no statistical difference for the synthetic datasets. The results of the GP with depth 6 are the best among all tested methods in all the 10 synthetic datasets.

As we are interested in model interpretability, Table 3.4 shows the complexity of the functions obtained by all methods. For the GP, the complexity is given by the number of nodes present in the trees of the returned individuals. For the linear regression methods, we report the number of coefficients different from 0.

Note that, for all datasets, the number of nodes of the GP trees when with maximum tree depth 2 and 3 are low, but the RMSE is high when compared to the version run with maximum tree depth 6. Also observe that the linear regression with L1, for most cases, returns a constant as the function that describes the data.

When a maximum tree depth of 6 is used, allowing for more complex models to be generated, the resulting number of nodes is high, which would preclude a level of interpretability suitable for the function to be understandable by a human being. Thus, from now on, our efforts are focused on locally approaching the curve obtained by the GP - D6 in order to have an interpretation of the model produced.

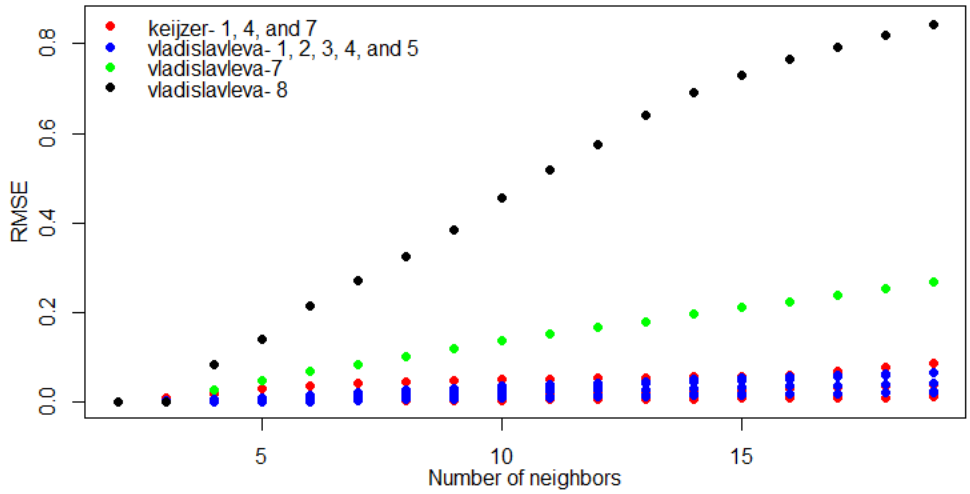
3.2.3 Quantitative Evaluation of ELA

ELA finds a local explanation to the model predictions based on the nearest neighbours of the point being predicted. This section tests the performance of the method and compares with the performance of the original functions obtained with all points.

Table 3.4: Complexity of the functions found by the methods, where for GP methods it is based on the average number of nodes present in the trees of the best individuals, and for LR methods it is based on the number of coefficients that are different from zero.

Dataset	Average size of the best individual			Non-zero coefficients		
	GP (D2)	GP (D3)	GP (D6)	LR	LR L1	LR L2
keijzer-1	4.9	13.6	67.9	0	0	0
keijzer-4	4.3	6.3	76.4	1	0	1
keijzer-7	7.0	15.0	78.5	1	1	1
vladislavleva-1	7.0	13.5	55.9	2	0	2
vladislavleva-2	3.9	7.2	67.7	1	0	1
vladislavleva-3	4.0	10.0	70.3	2	0	2
vladislavleva-4	6.4	11.1	50.0	5	0	5
vladislavleva-5	7.0	13.4	45.2	3	0	3
vladislavleva-7	7.0	14.8	75.5	2	0	2
vladislavleva-8	7.0	14.5	79.2	2	0	2

Figure 3.2: RMSE variation of the local explanations with different numbers of neighbors.



Source: created by the author.

Influence of the value of k: We first make an analysis of the impact of the single parameter the proposed method has on the synthetic datasets, which is the number k of neighbors in the training set that will be considered in the local linear regression. We varied the value of k from 2 to 19.

As observed in Figure 3.2, the smaller the number of neighbors the smaller the error, but the higher the chances of overfitting the model to a very small set of points and generating a "false" explanation. We understand that by increasing the number of neighbors, we take the risk of selecting instances that are far from the point of interest, increasing the error. In order to allow for a better generalization without significantly increasing the error, a value of k equals to 5 will be used in further experiments, as it

Table 3.5: Mean RMSE of neighborhood of test points.

Dataset	Real/GP	Real/ELA	GP/ELA
keijzer-1	0.039	0.057	0.029
keijzer-4	0.181	0.181	0.003
keijzer-7	0.128	0.130	0.006
vladislavleva-1	0.044	0.044	0.008
vladislavleva-2	0.144	0.144	0.005
vladislavleva-3	0.496	0.495	0.014
vladislavleva-4	0.128	0.128	0.000
vladislavleva-5	0.086	0.088	0.008
vladislavleva-7	1.607	1.607	0.045
vladislavleva-8	0.635	0.654	0.145
wineRed	0.610	0.610	0.000

represents a good trade-off between error and generalization. Note that in Chapter 4 we present an improved process for neighbor selection by making this parameter more dynamic in relation to the local density of the instance of interest.

Results of local approximation: Table 3.5 shows the results of the RMSE found when comparing the real values of the training points used as neighbours for the local approximation by ELA with the predictions found by GP and ELA, and also compares the values predicted by GP and ELA. Note that these results consider the average RMSE of the points used to generate the local approximation for all test points. For example, in a hypothetical scenario with 10 test points and k equals 5, we have an average RMSE over 50 different training points. The rationale behind these results is to evaluate the impact that local models have in the errors of the training points. Observe that, for most cases, the real versus GP and real versus ELA errors are very similar, showing both functions are not very different in those regions of the space. On the other hand, it does not make sense to calculate the errors on the test set, as the predictions are made by the original model produced by the GP, and only the explanation uses this local model.

As we can see, the RMSE average results between the real data and ELA are very close to the ones comparing the real data with the GP predictions. Additionally, the difference between the GP and ELA RMSE is considerably lower, in all cases, than the mean RMSE between GP and the real data. From that we can conclude that the proposed method is actually locally describing the behavior of the function obtained by the GP.

Observing the GP/ELA results when compared with the error of the linear regression approximation, we can also have an intuition of whether the provided explanation was good or not. Note that a small training set (from where neighbors are selected to construct the linear regression) hinders the approximation of the ELA explainer with the

Table 3.6: Wine: Complexity of the functions found by the GP methods.

Average size of the best individual		
GP (D2)	**GP (D3)**	**GP (D6)**
7.0	14.5	52.3

Table 3.7: WineRed: Mean RMSE of the GP models.

TRAINING SET			TEST SET		
Depth 2	**Depth 3**	**Depth 6**	**Depth 2**	**Depth 3**	**Depth 6**
0.724 (0.037)	0.672 (0.020)	**0.647 (0.013)**	0.739 (0.035)	0.691 (0.030)	**0.667 (0.016)**

model. In Table 3.5, the datasets with smaller training sets (keijzer-1 and vladislavleva-8) are among the three worst results of the approximation with the model.

Finally, we show an example of the results found by ELA for a one-dimensional dataset, keijzer-1, with different regression methods and using ELA. As previously mentioned, the function this dataset represents is defined in Equation 3.4. As previously reported, the GP returned tree with an average number of nodes of 65.8 when running with maximum tree depth 6. The linear regression for the same dataset, either by the common method or using L1 and L2, resulted in the following line:

$$LR(T) = -0.04397 \tag{3.6}$$

We chose the following point to explain the prediction given by the GP: $(0.647, -0.129)$. After selecting its 5 nearest neighbors, the function found was:

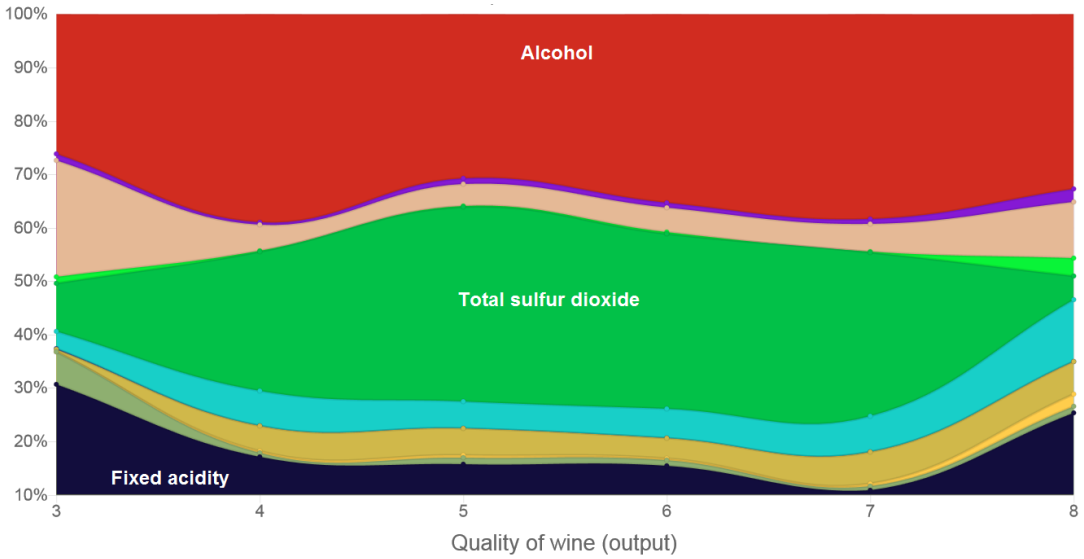
$$LR(NN) = -0.489x + 0.183 \tag{3.7}$$

Figure 3.1 shows that the local explanation we generate approximates well the equation obtained by the symbolic regression and is much simpler to interpret than the original function.

3.2.4 Qualitative case study: Wine dataset

Here we analyze the dataset *wineRed*. We chose this dataset for two main reasons. First, it presents a nonlinear relationship between the attributes and the output. Next, the subject of the dataset, which describes the characteristics of wines of the red type and

Figure 3.3: Global explanation for attributing quality to a wine. The x-axis represent the note the wine received, the y-axis the relative importance of the feature when that note is attributed to a wine.



Source: created by the author.

the output the quality, with notes varying between 0 and 10, can be interpreted with a low degree of expertise in the problem.

The set of attributes used to describe each instance in *wineRed* is as follows: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulfates, and alcohol.

Analogously to what was observed in the synthetic datasets, in *WineRed* there is a relationship between the growth of the maximum tree depth of the GP and the complexity of the model in terms of the average size of the best individual (Table 3.6) and smaller RMSE averages (Table 3.7).

Table 3.8 shows an example of explanation found by ELA in GP (D6), where we show the test point and the value of the variable to be predicted. In this case, for a wine receiving note 6, there is an error of 0.223 (6.453 for GP and 6.676 for ELA). The table lists, apart from the coefficients of the linear regression produced by ELA, the importance of the attributes in the analyzed test instance.

A relevant feature that we can highlight at this point is that the ELA tool provides an additional interpretability feature that is the importance of attributes for regression of the test instance.Looking at the values of attribute importance, we can highlight two: alcohol (75.9%) and fixed acidity (8.8%).

The last lines in Table 3.8 show the range of the values of the attributes of neighbors considering a maximum range of 10% of variance in the output. By verifying the modifications of the values of the attributes we can understand how wines similar to the presented could be manufactured. For example, the residual sugar attribute could range between 1.6 and 3.3.

Table 3.8: Information of the regression methods and ELA for the wineRed dataset.

Test point analyzed
= [10.8, 0.29, 0.42, 1.6, 0.084, 19, 27, 0.99545, 3.28, 0.73, 11.9]
Real value = 6.0
Value predicted by GP= 6.453
Value predicted by Local Explanation= 6.676
Coefficients of linear regression
= [-0.09, 0.32, 0.05, -0.17, -0.04, 0.03, -0.01, 0.02, 0.11, 0.18, 0.74]
Intercept= -1.8424176877057139
Importance of attributes
= [8.82, 0.79, 0.17, 2.39, 0.03, 5.46, 1.82, 0.14, 3.24, 1.17, 75.95]
Variation of attributes among neighbors within a maximum range of 10% of variance (6.4313 - 6.6138):
7.9 <= fixed acidity <= 10.8
0.2 <= volatile acidity <= 0.33
0.35 <= citric acid <= 0.42
1.6 <= residual sugar <= 3.3
0.054 <= chlorides <= 0.084
6.0 <= free sulfur dioxide <= 19.0
15.0 <= total sulfur dioxide <= 27.0
0.99458 <= density <= 0.99545
3.28 <= pH <= 3.32
0.73 <= sulfates <= 0.8
11.8 <= alcohol <= 12.0

Finally, since we have a large set of explanations for different test points, we can also provide an *overall explanation of the model* using a graph of stacked areas[2], as shown in the Figure 3.3, where the x axis represents the output of the method (in this case, the quality of the wine) and the y axis the relative importance of that attribute for a wine receiving that note. In this graph the attributes are sorted following the same order they are listed in Table 3.8: the first attribute is in the lower portion of the graph is fixed acidity in dark blue, the second is volatile acidity in dark green and so on, up to alcohol shown in red at the top of the graph area. Looking at this graph we observe that, regardless of the quality attributed to a wine, in general the most relevant attributes are alcohol (red) and total sulfur dioxide (green). The alcohol feature has an importance above 20% regardless of the quality of the wine. The Total sulfur dioxide feature is more important for intermediate quality wines, more than 30%, when compared to high and low quality wines. In addition, we also observed that the Fixed acidity and pH features have a behavior opposite to Total sulfur dioxide, that is, they are more relevant in the qualification of lower and upper wines and less important in intermediate wines.

[2]https://www.chartjs.org/

Sulfur dioxide is used in wine-production as a preservative due to its anti-oxidative and anti-microbial properties, but also as a cleaning agent for barrels and winery facilities. Studies have shown that wines may have their sensorial attributes deteriorated (e.g., oxidise) if the concentration of free sulfur dioxide falls below a particular critical level, specific for each particular wine. Also, wines with higher pH values may deteriorate at higher critical levels of free sulfur dioxide [43].

We can also note that the fixed acidity attribute is more important in the task of assigning notes for low and high quality wines, losing its importance among intermediate quality wines.

3.3 Summary

This chapter presented ELA, a method capable of generating interpretable explanations for the results provided by regression algorithms. Differently from other approaches previously presented in the literature, the proposed method uses the neighborhood concept of a certain test point of interest to carry out a local linear regression and identify how much each input attribute influences the output. The strategy adopted also provides ranges by which the attributes can be changed locally, bringing more information for interpretation. In addition, a graph-based view of stacked areas is proposed to provide an overview of the overall behavior of the model. The experiments showed that the explanations provided have a strong approximation with the results obtained by the symbolic regression method in terms of RMSE, besides providing useful explanations for understanding the results.

Chapter 4

DELA: Dynamic Explanation by Local Approximation

In this chapter, we propose an improved version of ELA, named DELA (Dynamic Explanation by Local Approximation). DELA tackles the main challenge of ELA, which despite its simplicity, could have its explanations enhanced by carefully defining its three main hyperparameters: i) the distance metric used to measure the similarity between instances (Euclidean distance); ii) the number of neighbors used to perform the local linear regression (5 neighbors are currently recommended); iii) the way to determine the importance of the features: calculated based solely on the instance of interest. Moreover, initial experiments showed the performance of ELA in non-normalized input data. This can make the explanations unstable for datasets with features of high variability. Hence, the impact of data normalization is also studied in this chapter.

DELA has the ability of better adapting to the characteristics of the dataset the regression model was created to. In sum, the main improvements added to ELA's core regard decisions that are now automatic and made according to each dataset, including:

(i) A more appropriate distance measure for each specific dataset is selected - Subsection 4.1.1.

(ii) The number of neighbors selected to provide the explanation is chosen automatically according to the local density of the instance of interest - Subsection 4.1.2.

(iii) Feature importance is calculated locally but considers the contribution to the output of the target instance provided by the neighborhood, rather than just considering the evaluated test instance. - Subsection 4.1.3.

In addition, in this chapter we perform a quantitative analysis of the proposed approach in a significantly larger of datasets: 10 synthetic datasets and 20 real benchmarks, with known non-linear relations between the predictive and target variables. To demonstrate the agnostic capability of DELA, we present the explanatory results of the output obtained by a Random Forest (RF) regression algorithm, while the initial proposal of the ELA explanation method used a genetic programming (GP) algorithm. Then, to fill the

gap of adding a human specialist to the loop, we perform a case study in a real-world medical dataset, where the task is to predict the level of severity of a patient in terms of the diagnosis of rheumatic heart fever (RHD).

Finally, the explanations provided by DELA were compared quantitatively and quantitatively with ELA and LIME. The results show that DELA is able to obtain local explanations with errors that are statically similar to ELA, with the advantage of being less affected by datasets with high variance in features and setting the value of different hyperparameters. Furthermore, DELA was able to qualitatively identify the importance of features underestimated by ELA, be less abrupt in global explanations by varying the model output and return semantically valid local patterns in its explanations – something that LIME, for instance, is not able to do.

The remainder of this chapter is organized as follows: Sect. 4.1 presents the proposed method of DELA, while Sect. 4.2 discuss the impacts of the proposed changes to the original method. In Sect. 4.3 presents the quantitative and qualitative experiments. Finally, Sect. 4.4 draws conclusions.

4.1 Proposed Methodology

This section introduces DELA (Dynamic Explanation by Local Approximation), an extended version of ELA that makes it adaptable to the characteristics of different datasets.

Let us assume we have a training set $T = p_i = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ and a test set $T' = p_j = \{(\mathbf{x}_j)\}_{j=1}^{m}$ — with $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$ for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$. Analogously to ELA, DELA is model-agnostic and can be used to explain the predictions y' of non-linear regression models. For that, DELA performs a local approximation with a linear regression method to generate explanations for each prediction y' generated by the original applied model.

Among the main drawbacks of ELA we mention: the use of a fixed number of neighbors chosen to perform the linear regression (currently we recommend using 5) at the time of the local explanation, (ii) the indiscriminate use of the Euclidean distance metric to compare the instances, which can be high-dimensional and suffer from the curse of dimensionality, (iii) consider only the evaluated instance to calculate the local features importance.

Each of these limitations will be tackled by DELA by taking into consideration the individual characteristics of each dataset. For that, we propose:

(i) A study of different distance measures: In the ELA method, a distance metric (more specifically, the Euclidean distance) is used to find the training instances closest to the instance to be explained. The use of an inappropriate distance measure can directly affect the quality of the explanation produced by regression models. Hence, we analyze different distance measures in Subsection 4.1.1 and discuss which is the most appropriate.

(ii) An adaptive choice of neighborhood for local explanation: Different strategies can be used to generate local explanations. Some methods, such as LIME, proposed using local perturbation around the test instance to generate fictitious instances that are close to the point of interest – which can generate semantically invalid points. In ELA, instances from the neighborhood of the instance of interest were selected, but the number of instances was fixed and set up by the user. This strategy can neglect regions where solutions are more spread out in the space, where neighbors may not be as close as desired. In this case, using a small number of neighbors can be appropriate, even increasing the risk of overfitting. On the other hand, being in a dense region, a larger number of neighbors can help decrease the local error of the model. Hence, we propose a method to automatically set the value of k according to the local density of the neighborhood of the evaluated test point, as discussed in Subsection 4.1.2.

(iii) The importance of the features of the problem: The ELA method measures the importance of features by taking into account the proportional contribution of each linear regression coefficient to the output of the instance of interest. For DELA, we propose that the importance value should be considered locally, not pointwise. Thus, the new feature importance calculation will take into account the feature averages of all instances selected as neighbors in the local explanation of a particular instance. This means that feature importance will be evaluated not only in the instance in question but also in other nearby instances that may influence the local explanation of the model. This subject is discussed in Subsection 4.1.3.

Further, we show that by not normalizing the input data, the explanations provided by ELA become more susceptible to bias embedded by features with different scales. Subsection 4.2.1 presents a data normalization process that should be used as a preprocessing method for both ELA and DELA.

4.1.1 Instance distance measures for regression

For ELA to work and provide explanations compatible with the regression performed, a suitable distance metric for finding the neighbours of a point is of great importance. The growth of the dimensinality of datasets can make it trick for traditional distance metrics, such as the Euclidean distance, to capture the actual distances between instances due to the curse of dimensionality. ELA used the Euclidian distance (Eq. 4.1) to find neighbors, where d is the number of input features.

$$dist(x, x_i) = \sqrt{\sum_{a=1}^{d}(x_a - x_{ia})^2} \tag{4.1}$$

Given its drawback, this section proposes to evaluate other distance metrics which can better capture distances in high dimensional spaces.

There are a few studies in the literature that evaluate which distance/similarity metrics are most appropriate for high-dimensional data, but generally these studies are performed for classification or clustering problems. [5], for instance, evaluated the behavior of the Ls norm (also known as the Minkowski distance) and showed how the results can vary when we modify the value of s. After observing that smaller values are more appropriate in high dimension data, e.g., the $L1$ norm (Manhattan distance) being preferred over the $L2$ norm (Euclidean distance), fractional values were tested. They observed that fractional values of s improve the effectiveness of clustering algorithms.

Both [61] and [42] have also evaluated distances in the context of microarray data for clustering and classification tasks. In the former, 15 different distance metrics were evaluated for 52 datasets. They showed that the appropriate metric depends directly on the scenario in question. In the latter, the authors also mention that there is no consensus in the literature about which metric works best. After experiments with Hierarchical and K-means clustering algorithms they found that the distances of Minkowski, Cosine, and Pearson's correlation were the best options for their context.

Here, to address regression problems, appropriate measures are required. Therefore, we evaluate the effectiveness of 8 different distance/similarity measures, including a parameter variation in the Minkowski measure that will result in 30 alternatives. For all measures, u and v are instances of the datasets, and d corresponds to the number of input features. The measures being considered are:

(i) Minkowski:

$$dist(u, v) = (\sum_{a=1}^{d}|u_a - v_a|^s)^{1/s}, \tag{4.2}$$

where we evaluated values of s in the interval [0.1,3.0] in steps of 0.1.

(ii) Cosine

$$dist(u,v) = 1 - \frac{u \cdot v}{||u|| \times ||v||}, \tag{4.3}$$

where $||x||$ is the length of the vector x defined by $\sqrt{\sum_{i=1}^{n} {x_i}^2}$ and $u \cdot v$ is the dot product of u and v.

(iii) Correlation

$$dist(u,v) = 1 - \frac{(u - \overline{u}) \cdot (v - \overline{v})}{||u - \overline{u}||_2 ||v - \overline{v}||_2}, \tag{4.4}$$

where $\overline{u}$ and $\overline{v}$ are the mean of the elements of u and v and, again, $u \cdot v$ is the dot product of u and v.

(iv) Bray-Curtis

$$dist(u,v) = \frac{\sum_{a=1}^{d} |u_a - v_a|}{\sum_{a=1}^{d} |u_a + v_a|}. \tag{4.5}$$

Note that the Bray-Curtis distance is in the range [0, 1] if all coordinates are positive, and is undefined if the inputs are of length zero.

(v) Canberra

$$dist(u,v) = \sum_{a=1}^{d} \frac{|u_a - v_a|}{|u_a| + |v_a|} \tag{4.6}$$

when u_a and v_a are 0 for a given i, the fraction $0/0 = 0$ is used to avoid undefined values.

(vi) Chebyshev

$$dist(u,v) = max_a |u_a - v_a| \tag{4.7}$$

(vii) City Block (Manhattan)

$$dist(u,v) = \sum_{a=1}^{d} |u_a - v_a| \tag{4.8}$$

(viii) Standardized Euclidean

$$dist(u,v) = \sum_{a=1}^{d} |u_a - v_a|^2 \tag{4.9}$$

Note that we are expanding the possibilities for distance measures by relaxing the properties that a distance metric must have, i.e., non-negativity, identity, symmetry, and the triangle inequality By relaxing these properties, we can consider different ways of measuring the distance between points. These measures may be more suitable for certain types of complex data, such as high-dimensional data or data that do not follow a normal distribution. However, it is important to remember that by relaxing the properties of the distance metric, we may introduce distortions in the data analysis and influence the results obtained. Therefore, our proposed approach will individually choose the most appropriate measure for each dataset.

The choice of the metrics listed above rely on the fact that they have been previously shown to be effective in various contexts in the literature. Specifically, Minkowski measures with fractional parameters can be suitable for high-dimensional data, particularly when the instance density is low relative to the number of dimensions. This allows for capturing subtle changes between instances. We will vary the Minkowski metric parameter from 0.1 to 3.0. This parameter controls the degree of sensitivity of the metric to differences in feature values, with lower values giving more weight to small differences and higher values giving more weight to large differences.

The cosine metric is traditionally used for vector comparison, while correlation may be useful for evaluating the relationship between independent variables and the dependent variable. The Bray-Curtis metric is particularly useful when evaluating the dissimilarity between the proportions of features in two different instances, while the Canberra metric is useful for sparse data.

The Chebyshev metric, in turn, is a measure of maximum distance and is useful for data with outlier values. The City Block metric considers only absolute differences between values and performs well when the relationship between instances is linear. Finally, the standardized Euclidean metric is suitable for data with different variances, as it takes into account standardized differences between variables. By considering a diverse set of metrics, we can explore different aspects of the data and evaluate which metric is best suited for a particular regression model.

Our evaluation approach starts with the assumption that instances with close output values must have relatively close predictive features. Hence, the proposed distance selection method assumes the existence of linearity between the model output for each instance and the distances that separate these instances. The distance measure that presents greater correspondence between the input features and the output feature is considered the best to be used in the specific dataset D.

Algorithm 2 presents the proposed distance measure selection method. The algorithm receives as input a training set T, composed of a matrix of features X and an output vector Y, and a list of distance measures to be evaluated LM. In line 1, the output set is sorted in ascending order and the indices of these instances are stored in

Algorithm 2 Distance measure selection for dataset D.

Require: T (training set), LM (list of distance measurements to be evaluated)
1: $Y_{idxSort}$ = Sort T_Y in ascending order
2: $I_{y-smlest}$ = returns instance of lowest output($Y_{idxSort}$, I)
3: $measurementCounters$ = [ ]
4: **for** each M ∈ LM **do**
5: $Distances$ = calculates distances between $I_{y-smlest}$ and all others in $I(M)$
6: $Distx_{idSorted}$ = Sorts $Distances$
7: $counter$ = 1
8: **for** each p ∈ $Y_{idxSort}$ **do**
9: aux = 0
10: **for** each p' ∈ $Distx_{idSorted}$ **do**
11: **if** p == p' **then**
12: $counter$ += aux
13: Remove p' from $Distx_{idSorted}$)
14: **else**
15: aux += 1
16: **end if**
17: **end for**
18: **end for**
19: $measurementCounters$.append($counter$)
20: **end for**
21: **return** $LM(min_{idx}(measurementCounters))$

$Y_{idxSort}$. In line 2, the lowest output instance (top of the ranking) is identified and stored in $I_{y-smlest}$. In line 3, an empty list is created to store the result obtained by each distance measure that will be evaluated. In the loop between lines 4 and 20, we will individually assess the pertinence of using each of the distance measures in that specific dataset. Line 5 calculates the distance, according to the measure being evaluated (M), between I_{smlest} and all others in T_X. Next, these distances sorted by ascending value and their indices are stored in $Distx_{idSorted}$. This step is followed by a comparison between the vectors $Y_{idxSort}$ and $Distx_{idSorted}$, made in the loop between lines 8 and 18. In the loop, for each point in $Y_{idxSort}$, we count how many positions are accessed in $Distx_{idSorted}$ until we find it. When found, the point is removed from $Distx_{idSorted}$ (line 13). The rationale behind this approach is to compare the order of the outputs regarding the inputs. With our linearity assumption, the perfect distance would generate 0, i.e., both rankings in $Distx_{idSorted}$ and $Y_{idxSort}$ would be equal. In line 19 we store in $measurementCounters$ the counter obtained by using the distance measure evaluated in that iteration. We can then say that the distance measure with the lowest value (line 21) is more appropriate for regression problems for the dataset being analyzed.

4.1.2 A heuristic for choosing the number of neighbors

As already mentioned, ELA works by finding the k nearest points of each interest point p to perform a linear regression on that neighborhood and then get a local interpretation of the prediction.

In the initial version of ELA, the value k was a user-defined parameter, and in order to get an idea of the influence of the number of neighbors in the results of a local linear model, we performed experiments varying the values of k from 2 to 19. These experiments showed that the smaller the number of neighbors the smaller the error, but the higher the chances of overfitting the model to a very small set of points and end up generating a "fake" explanation. After preliminary analysis in ELA, k was set to 5, as it represented a good balance between error and generalization.

In this section we propose a way to automatically find the value of the neighborhood size k for each point being analyzed according to the density of the region it belongs to. The heuristic proposed is based on the fact that data is not usually equally distributed in the instance space. The proposed heuristic for defining variable neighborhoods is presented in Algorithm 3, and was inspired by the clustering algorithm DBSCAN [34]. The algorithm receives as input the set of training data T, the test point t we want to find the neighborhood for and the parameters $MinP$ – which represents the minimum number of points that will be accepted as input for the local linear regression, and $MaxD$, which represents the maximum acceptable distance between neighbors.

Note that by defining the minimum number of accepted neighbors ($MinP$), we aim to avoid overfitting. The fewer the instances, the higher the chance of the regression to overfit and fail to capture the true relationship between features. Moreover, the lack of data can lead to greater variability in the results, making it difficult to evaluate the accuracy of the model. By defining the maximum acceptable distance to consider points as neighbors ($MaxD$), we avoid including instances too different in the neighborhood, which may not be locally relevant. Additionally, including instances that are too far away can increase variability and negatively affect the accuracy of the local model. Finally, limiting the maximum distance can also help reduce the processing time for obtaining explanations.

In line 1, the distances between the test point t and all training points $p \in T$ is calculated. These points are then sorted in ascending order according to their distance to t (line 2.) In the loop between lines 4 and 13, for each ordered point distance, we check whether it respects the limit imposed by MaxD (line 5). If yes, the point is considered as a neighbor (line 6). If not, we check if the minimum number of neighbors has already been reached (line 8). If the minimum has not yet been reached, then MaxD value is redefined in line 9 and the training point is considered as a neighbor (line 10).

Algorithm 3 Local definition of k: variable neighborhood.

Require: T (training set), t (test point), $MaxD$, $MinP = 5$
1: $Distances$ = vector of pairs $< p \in T, dist(p, t) >$
2: $Distances_{sorted}$ = sort points in $Distances$ according to $dist$
3: $neighbors$ = [ ]
4: **for** each point p $\in$ $Distances_{sorted}$ **do**
5: **if** p.dist $<$ $MaxD$ **then**
6: $neighbors$.append(p)
7: **else**
8: **if** length($neighbors$) $<$ $MinP$ **then**
9: $MaxD$ = p.dist
10: $neighbors$.append(p)
11: **end if**
12: **end if**
13: **end for**
14: **return** $neighbors$

In this problem we consider MinP with a value of 5, as it was previously found by ELA to represent a good balance between error and generalization. In addition, if values close to the minimum are chosen by the method only for local region with few instances, i.e., a low density region, this can help avoiding overfitting.

The value of MaxD, on the other hand, was chosen according to the heuristics adopted for defining the parameters of DBSCAN [100], and we can use: i) the median of the distance between the two closest neighbors of all training points; ii) the third quartile of the distance of the two closest neighbors of all training points; iii) the maximum distance between the two closest neighbors of all training points.

4.1.3 Feature Importance in Local Linear Regression

We define feature importance as the contribution that each feature exerts to the final value of the output of an example p, given by y'. Intuitively, the initial version of ELA used the values of the coefficients returned by the local regression multiplied by its normalized value in p. This value is expressed as a percentage of importance, taking into account the importance of all features, as shown in Equation 4.10.

$$Importance_{x_i} = \frac{|coefficient_{x_i} \times x_i| \times 100}{\sum_{a=1}^{d} |coefficient_{x_a} \times x_a|} \tag{4.10}$$

There are studies that have already proposed several metrics for measuring feature importance from the values of the returned coefficients with different, being the simplest

and most straightforward way to look at the values of the coefficients, which has a big problem since they are not invariable in scale. Other approaches with different levels of complexity have been proposed [46, 2, 115], such as to use the model's variance [46].

Here we consider that identifying the most important features for the *local* approximation performed, i.e., features of locally selected neighbors, is the key point. For similar contexts, where the level of importance is not sample-independent, [2] first introduced the importance level metric:

$$LevelImportance_i = b_i \times mean(V_i) \tag{4.11}$$

where b_i is the coefficient returned for feature i and V_i the vector of values found in the neighboring examples for feature i. This metric was more recently revisited in the works of [54] and [46].

In addition, we want this metric of importance to reflect the relevance that a given feature has in the predicted value of a local region in relation to other features. Thus, we look at the importance module of each feature and normalize it (Equation 4.12), making it into a percentage.

$$Importance_{x_i} = \frac{|LevelImportance_i| \times 100}{\sum_{a=1}^{d} |LevelImportance_a|} \tag{4.12}$$

4.2 Experimental Setup

In this section we show the results obtained in the application of the proposed improvements to the ELA explanation method in order to build DELA. The dataset used in the experiments is described in Table 4.1, which shows the total number of features (predictive and target (Attributes), Number of data points used in the training (Train) and test (Test) phases. In total there are 10 synthetic and 20 real-world dataset. All synthetic datasets were originally presented in [67]. The real datasets are publically available from UCI and Kaggle [28, 20, 22].

Table 4.1: Dataset used in experiments involving internal improvements in feature-based explanation methods.

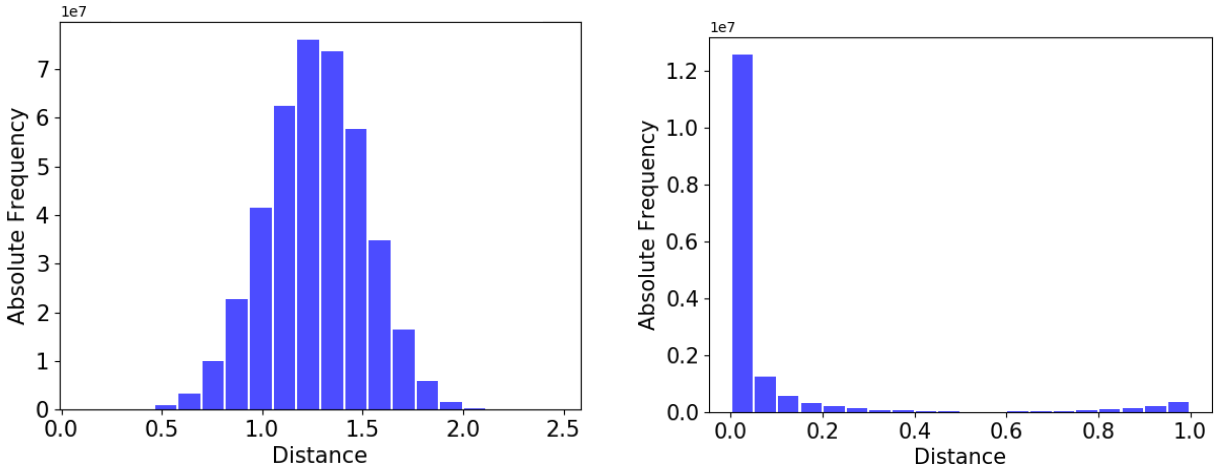
Id	Dataset	# Attributes	# Train	# Test	Nature	Variance
1	keijzer-1	2	1,415	607	Synthetic	0
2	keijzer-4	2	141	61	Synthetic	0
3	keijzer-7	2	764	327	Synthetic	0
4	vladislavleva-1	3	1,487	638	Synthetic	0
5	vladislavleva-2	2	225	96	Synthetic	0
6	vladislavleva-3	3	3,978	1,705	Synthetic	0.001
7	vladislavleva-4	6	4,217	1,807	Synthetic	0.002
8	vladislavleva-5	4	2,100	900	Synthetic	0.045
9	vladislavleva-7	3	910	390	Synthetic	0
10	vladislavleva-8	3	797	342	Synthetic	0
11	cholesterol	14	209	88	Real	2630.956
12	auto-mpg	8	279	119	Real	9.29e+5
13	sensory	12	403	173	Real	0.231
14	strike	7	437	188	Real	5.32e+5
15	day_filter	12	510	221	Real	2.910
16	house	80	1,022	438	Real	1.33e+6
17	wineRed	12	1,279	320	Real	115.431
18	abalone	8	2,924	1,253	Real	0.329
19	wineWhite	12	3,918	980	Real	1459.797
20	cpu_act	22	5,734	2,458	Real	7.62e+10
21	bank32nh	33	5,734	2,458	Real	2.585
22	puma32H	33	5,734	2,458	Real	0.450
23	compactiv	21	5,734	2,458	Real	6.86e+10
24	tic	85	6,876	2,947	Real	13.061
25	ailerons	41	9,625	4,125	Real	16.621
26	elevators	19	11,619	4,980	Real	59.350
27	california	8	14,448	6,192	Real	5.21e+5
28	house_16H	17	15,949	6,835	Real	1.11e+5
29	fried	11	28,538	12,230	Real	0
30	mv	10	28,538	12,230	Real	1.11e+5

4.2.1 Input data normalization

Both DELA and ELA were conceived to be agnostic explanation methods, and we know some algorithms are very susceptible to feature scaling. A large variance between the different features that describe the instances can cause bias, such as disregarding values on a smaller scale in the prediction process.

Internally, the method of explanation can also be subject to the same phenomenon, since from the nearest neighbors of a point of interest a linear regression is performed, and feature importance measures are derived from the model.

Figure 4.1: Histogram with distances to the instances.



(a) Fried dataset (small variance) (b) CPU_ACT dataset (large variance)

Source: created by the author.

In order to avoid biases that may be embedded both during the learning process of the regression model and in the explanation method, we propose that input data for DELA always undergo a normalization process. Therefore, before the model training process, all data will be normalized by the algorithm known as z-score, as shown in Equation 4.13, which accounts for the mean and standard deviation of the values for each feature.

$$z = \frac{x - mean(x)}{stdev(x)} \tag{4.13}$$

In order to better understand the process of input feature normalization (or its absence) on explanation methods, and to validate its relevance, we performed the following analysis on the datasets: we calculated the median value of each feature among all instances of training the same dataset, and measured the variance of values among the different features. The results obtained are shown in the last column of Table 4.1. Observe that, in our datasets, there are both cases with small and large variance. The latter represents the cases where the normalization process is more essential.

In a complementary way, the histograms in Figure 4.1 show the distances between all points of the same dataset, combined 2 by 2. Figure 4.1a represents a small variance dataset and Figure 4.1b a large variance dataset, and these histograms are similar for all other datasets.

Observe that datasets with lower variance have a behavior close to a normal distribution, making it simpler to distinguish between instances and, consequently, to choose the closest neighborhood. Datasets with a lot of variance tend to lose the significance of similarity, and the process of distinguishing between instances becomes possible only after the normalization process.

In another aspect, we can also look at how much a regression model can be affected

Table 4.2: Impact of the dataset normalization process on the RMSE of the model trained with a Random Forest regressor.

Dataset	RMSE	
	Normalized	Non-normalized
cholesterol	**1.03**	55.85
auto-mpg	**0.38**	2.74
sensory	0.93	**0.75**
strike	**0.89**	503.04
day_filter	**0.37**	741.58
house	**0.41**	3,3683.99
wineRed	0.87	**0.70**
abalone	**0.73**	2.35
wineWhite	0.79	**0.70**
cpu_act	**0.15**	2.69
bank32nh	0.77	**0.09**
puma32H	0.27	**0.01**
compactiv	**0.14**	2.56
tic	1.11	**0.26**
ailerons	0.43	**0.00**
elevators	0.52	**0.00**
california	**0.45**	51,529.31
house_16H	**0.62**	32,885.45
fried	**0.28**	1.40
mv	**0.01**	0.09

by the normalization of the data. The results shown in Table 4.2 report the Root Mean Squared Error (RMSE) obtained by a Random Forest algorithm for the real datasets. Note that the predictions obtained for the test set have a lower error for normalized datasets in 12 of the 20 analyzed sets, and higher in 8 datasets. This justifies that fact that not normalizing data in this context is not a good practice.

Finally, we also want to verify how much the model's explanation is also influenced by data normalization. For that, we compare the local explanations obtained by ELA, LIME and DELA.

Table 4.3 shows the intersection of the 5 most important features found for each instance of the test set, considering the normalized and non-normalized datasets. We selected 4 datasets to demonstrate this result: 2 datasets of small variance (Fried and Sensory) and 2 datasets of large variance (Cpu_atc and House).

In addition, we also analyzed the outputs obtained by the models when we normalized and did not normalize the input data. We aimed to identify the datasets where normalization resulted in significant differences in the model's output. To conduct this comparison, the outputs of the models using normalized data were reprocessed to revert to the original data scale. Subsequently, we calculated the square root of the sum of dif-

Table 4.3: Feature Intersection and differences obtained in instance outputs considering normalized and non-normalized datasets.

	Feature Intersection			**Normalization Difference**		
	LIME	**ELA**	**DELA**	**LIME**	**ELA**	**DELA**
Fried	5	3	3	0.0	0.5	0.4
Sensory	3	3	3	0.2	0.3	0.4
House	4	0	0	3,115.4	19,586.1	16,987.6
Cpu_act	4	2	1	1.1	4.1	3.8

ferences between the two outputs obtained (considering normalized and non-normalized inputs) for each instance, as described in equation 4.14. The results are presented in Table 4.3.

$$normalizationDifference = \sqrt{\frac{1}{|T'|}\sum_{i=1}^{|T'|}(\hat{y}_i - \overline{\hat{y}_i}))^2} \tag{4.14}$$

where $\overline{\hat{y}_i}$ is the model output with the data without normalization and $\hat{y}_i$ is the model output with the data normalized (back to original scale).

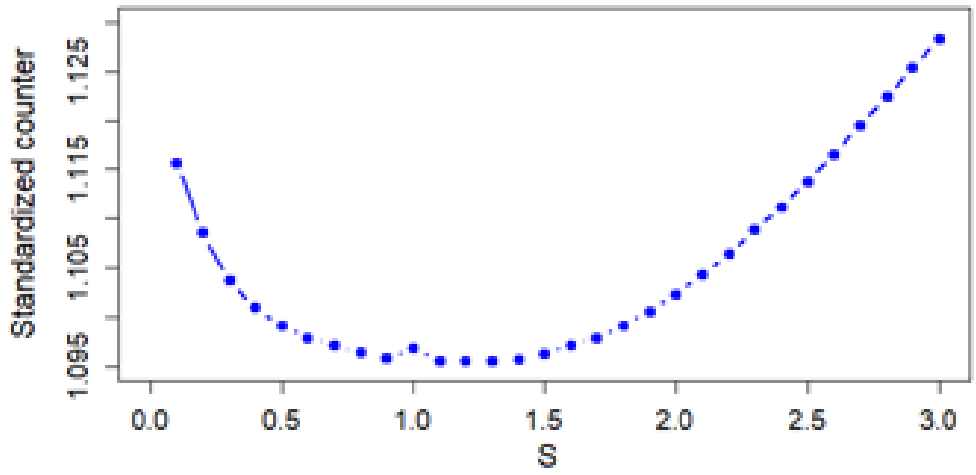
Observe that the impact of the normalization process is smaller in small variance datasets but can be huge in large variance datasets. The method that suffer less is LIME, as it generates 5,000 random fictitious neighborhood points for each test instance to be explained before performing the linear regression. ELA and DELA methods, in turn, select examples from the training set, and hence have, in general, a smaller number of neighbors for each point. Because of that, these methods are more susceptible to small changes in neighborhoods, and data normalization can completely change the results of model explanations.

4.2.2 Distance Measure Evaluation

Having established that the proposed method needs to run over normalized data, we start looking at the components we want to improve. First, we focus our attention on the distance measures, starting with Minkowski and its parameters.

We run Algorithm 2 with the Minkowski distance and different values of s for all normalized datasets. Recall that we varied the parameter s in the range [0.1, 3.0] with steps of 0.1, where s controls the degree of sensitivity of the metric to differences in feature values.. Algorithm 2 returns a count for each evaluated variation of the parameter s in the Minkowski measure. We normalized this count, where the best result is considered as

Figure 4.2: Result for Minkowski measure: heuristic counter mean.



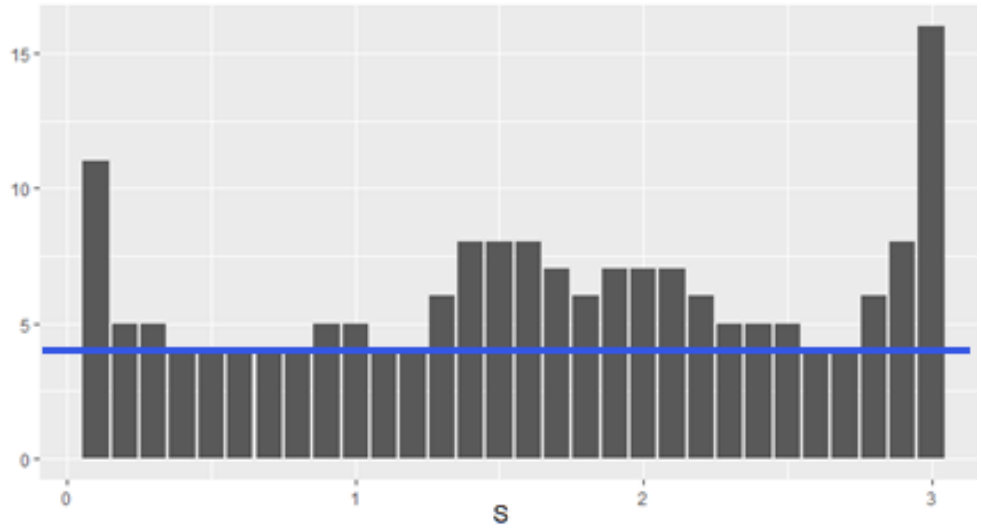
Source: created by the author.

1 and the others in the list receive a value proportional to 1. We executed Algorithm 2 for all variations of the parameter s across all 30 datasets listed in Table 4.1. The graph in Figure 4.2 shows the average of the normalized counts for each value of s across all datasets. As we can observe, the best values were obtained when s takes the values of 1.1 and 1.2, with a value of 1.095. When we consider the 30 observations using these two values of s, we found that the standard deviation was lower when the parameter was set to 1.1. Thus, in this context, setting s to 1.1 would represent the best solution with less variability.

After that, for the Minkowski measure, we also calculated how many times each possible value for s achieved the best result across all 30 datasets. In this analysis, the best value found for s was 3.0, as shown in Figure 4.3. Note that in some datasets, more than one distance measure achieved the best result. For instance, in datasets with only 2 attributes (keijzer-1, keijzer-4, keijzer-7, and vladislavleva-2), all measures performed equally well (the blue line in the plot was included for ease of visualization when excluding scenarios where the variation of the parameter s did not modify the results).

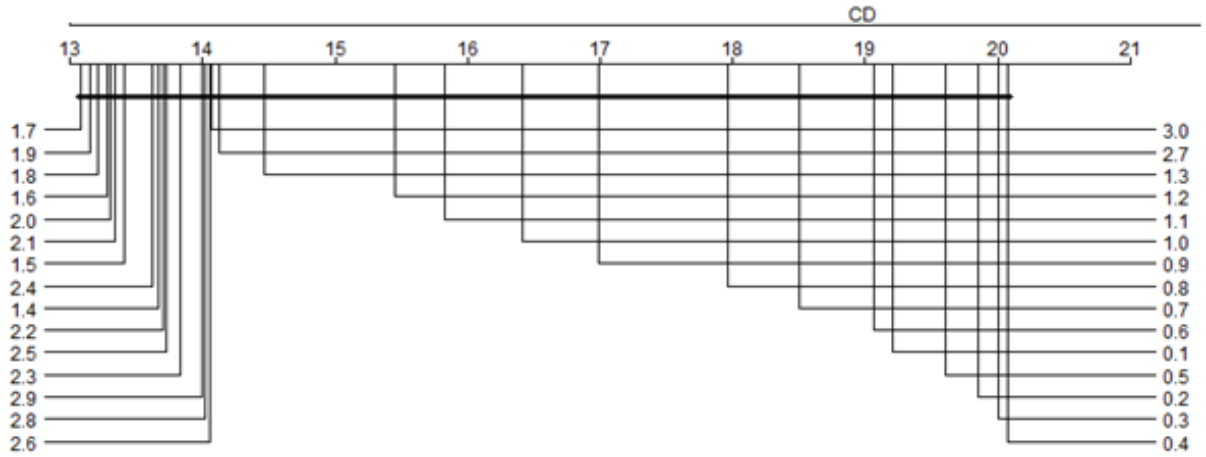
Finally, we plot a critical diagram from a Friedman test followed by a Nemenyi post-hoc with a usual significance level of 0.05 and Bonferroni correction. In these diagrams, the main line shows the average ranking of the methods, i.e., how well one method did when compared to the others. This ranking takes into account the absolute value obtained by each method according to the evaluated metric. The best methods are shown on the left (lowest rankings). The critical difference interval (CD in the plots) is determined by the Friedman test according to the significance level defined. If the average difference between two algorithms is greater than CD, then the null hypothesis that the algorithms have the same performance is rejected. Finally, the diagram connects groups of methods that do not present statistically significantly differences. Note that the size of the lines connecting the methods corresponds to the size of the CD interval.

Figure 4.3: Number of times the Minkowski measure with value s obtained the best result for the heuristic counter. The blue line indicates that in 4 datasets, all variations of the Minkowski measure obtained the same results.



Source: created by the author.

Figure 4.4: Result for Minkowski measure: critical diagram.



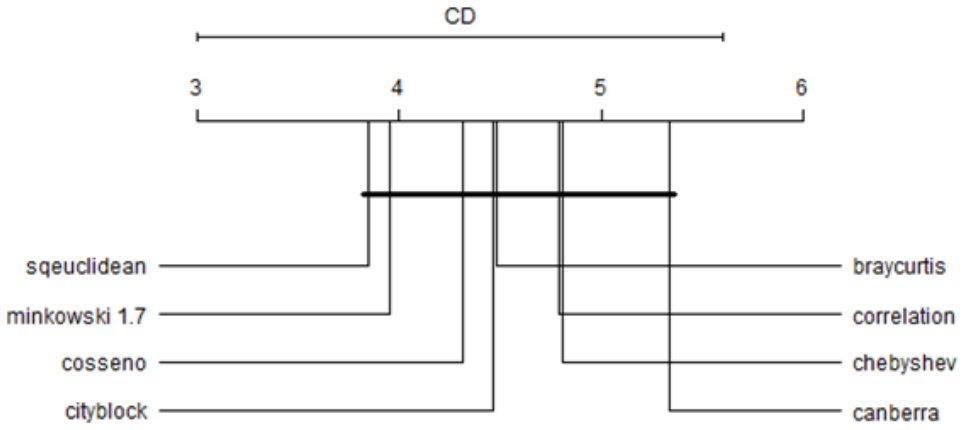
Source: created by the author.

This showed that none of the values tested for parameter s is statistically superior. The parameter with the best numerical result was s with a value of 1.7, as observed in Figure 4.4.

Now we comparing the Minkowski measure ($s = 1.7$) with the other measures listed above. The evaluation procedure is similar to the previous one. We define the best performer for each measure (across all datasets) as 1 and scale the other values proportionally. When plotting the critical diagram we found that numerically the best result would be obtained using the Euclidean distance metric, as shown in Figure 4.5. But, when checking how many times each of the measures managed to obtain the best result, we found that the best was the Canberra metric, as shown in Figure 4.6.

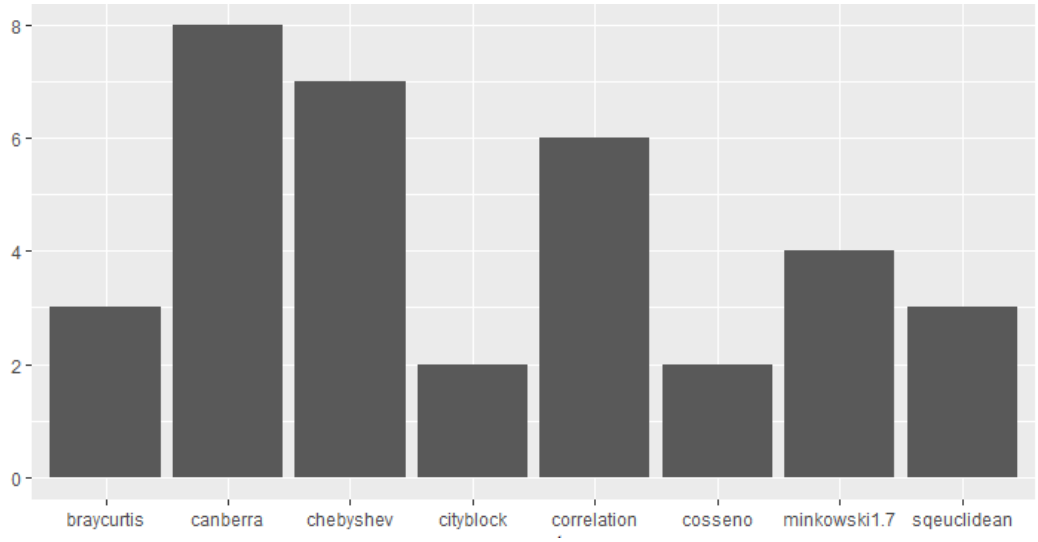
It can be concluded that there is no universally superior measure, as the best one

Figure 4.5: Result for distance measure: critical diagram.



Source: created by the author.

Figure 4.6: Result for distance measure: times that got the best result for the heuristic counter.

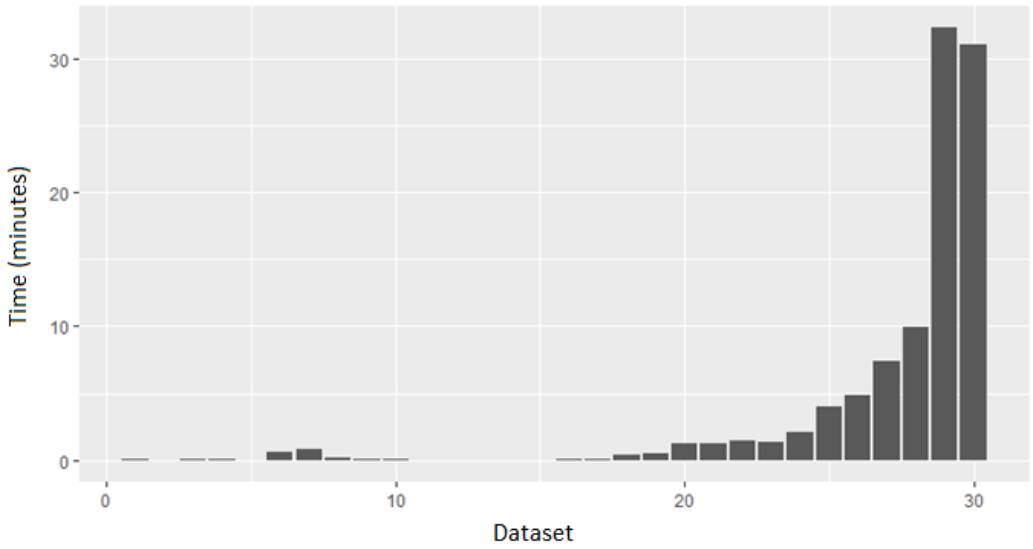


Source: created by the author.

varies depending on the dataset in question. Therefore, selecting a suitable measure is crucial to effectively discriminate among instances for a given dataset.

After that, we verified the time it would take to specifically choose the best measure through the proposed heuristic, in each of the datasets that used in this work. The results are shown in Figure 4.7. These experiments were performed on a computer with the following configurations: Intel(R) Xeon(R) E5620 2.40GHz, with Ubuntu 16.04 LTS operating system. As we can see, the heuristic is simple and takes only a few minutes to run, even on the largest datasets. Thus, we consider that the best strategy is for DELA to choose, prior to the explanation process, which distance measure is more appropriate for each specific dataset, once this process needs to be executed only once and then reused in new executions.

Figure 4.7: Result for distance measure: time needed (in minutes) to find the best distance measure, according to the proposed heuristic.



Source: created by the author.

4.2.3 Number of neighbors

To define the value of MaxD, we evaluated the three heuristics mentioned in Section 4.1.2, namely: i) the median of the distance between the two closest neighbors of all training points (Median); ii) the third quartile of the distance of the two closest neighbors of all training points (Q3); iii) the maximum distance between the two closest neighbors of all training points (Max). The results in terms of the minimum, median, and maximum number of neighbors selected for each instance in each dataset are presented in Table 4.4.

Observe that when using the median of the distances between the two nearest neighbors (first three column in the table), the instances tend to have the smallest number of neighboring points allowed by the proposed heuristic for use in local linear regression, that is, 5 neighbors. This is not appropriate, as regardless of the number of available instances or the size of the dataset, the chosen neighborhood value tends to be fixed, both for the minimum and median values. If we observe the maximum value of the neighborhood size determined for the instances using this strategy, we can obtain larger values, but even this maximum value is sometimes still smaller than the number of features o the dataset. This phenomenon occurs in 9 out of the 20 real datasets. Having a smaller number of instances to create the local linear regression compared to the number of features can make it difficult to identify the relationships between the independent variables and the dependent variable, and thus impair the local model's ability to provide good explanations.

When applying the heuristic of selecting the maximum distance between the two

Table 4.4: Minimum, Median and Maximum number of neighbors chosen when we vary the heuristic to determine the maximum distance that should be used to choose the neighborhood.

Dataset	Number of neighbors selected when using:								
	Median distance			Q3 distance			Max distance		
	Min	Median	Max	Min	Median	Max	Min	Median	Max
keijzer-1	5	5	6	5	5	9	156	700	714
keijzer-4	5	6	6	5	6	6	5	16	22
keijzer-7	5	5	7	5	5	7	5	7	11
vladislavleva-1	5	5	5	5	5	11	5	184	257
vladislavleva-2	5	5	6	5	5	6	5	86	104
vladislavleva-3	18	275	1,157	18	275	1,157	18	275	1,157
vladislavleva-4	5	5	15	5	5	38	5	385	674
vladislavleva-5	5	6	12	5	6	12	5	13	29
vladislavleva-7	5	5	6	5	5	8	5	13	27
vladislavleva-8	5	5	6	5	5	6	5	45	53
cholesterol	5	5	19	5	5	35	5	49	97
auto-mpg	5	5	6	5	5	12	8	53	107
sensory	5	5	8	5	5	8	5	7	14
strike	5	5	6	5	5	13	6	66.5	151
day_filter	5	5	8	5	5	14	5	27	64
house	5	5	44	5	6	81	5	150	302
wineRed	5	5	10	5	5	33	7	169	331
abalone	5	5	56	5	5	104	5	2,896	2,922
wineWhite	5	5	30	5	6	52	22	3,427	3,428
cpu_act	5	5	424	5	9	542	8	321.5	1,758
bank32nh	5	5	74	5	5	193	19	4,501	5,628
puma32H	5	5	7	5	5	8	5	13	26
compactiv	5	5	443	5	8	584	5	236.5	1,304
tic	5	7	313	5	14	636	10	6,870	6,872
ailerons	5	5	213	5	9	567	179	5,219	6,579
elevators	5	5	134	5	6	488	11,618	11,618	11,618
california	5	5	72	5	6	194	30	14,447	14,448
house_16H	5	5	3,737	5	167	6,114	14	12,353	14,927
fried	5	5	9	5	5	15	5	34	84
mv	5	5	11	5	5	19	5	96	714

nearest neighbors, the resulting instances tend to have a very large number of neighbors (varying from 7 to 14.447). Note that in 7 datasets, the maximum number of neighbors selected for an instance was very close to the total number of instances in the training set (abalone, wineWhite, bank32nh, tic, elevators, california, and house_16H). As a result, instances far from the location of the test point are used for explanation, which is also not ideal.

A better balance is obtained by the heuristic that chooses as neighborhood points those that are less than the third quartile of the two closest neighbors. In this case, we manage to obtain, for 12 datasets, a median value that exceeds the minimum and, at the

same time, we prevent the neighborhood from getting too large and distort the locality of the explanation. In this way, this will be the heuristic used to choose the neighborhood of the DELA explanation method.

4.3 Results

This section evaluates the explanations provided by the proposed method (DELA) and compare them to the explanations obtained by previously proposed explanation methods in the literature (ELA and LIME).

We will begin by evaluating the errors obtained in the task of predicting the output using regression models trained with the following algorithms: Linear Regression, Symbolic Regression with Genetic Programming, and Random Forest Regression (Section 4.3.1). The datasets used are described in Table 4.1. We show that inherently interpretable models yield inferior results in terms of error when compared to models built by black-box algorithms, justifying the need for explanation methods.

Next, we will conduct a quantitative (Subsection 4.3.2) and qualitative (Subsection 4.3.3) analysis of the explanations proposed for each of the datasets. Finally, to validate the explanations based on human domain experts' knowledge, we will conduct a case study on a real-world medical dataset (Subsection 4.3.4), where the task is to predict the severity level of a patient in terms of Rheumatic Heart Disease (RHD) diagnosis.

4.3.1 Regression with all points

In this section, we perform an analysis using different regression algorithms with all training and test points, considering both RMSE and model complexity. This is important for two reasons: first, to show we need more than a linear regression to solve the problem; second, to ensure the results of the local explanations will not increase the test error substantially.

As DELA is model agnostic, we used two regression methods considered state of the art: i) Symbolic regression with genetic programming (GP), implemented using the Python package Deap [37]; and ii) Random Forest Regressor (RF), found in the Scikit Learn library[1]. The parameters for these two methods were defined as follows.

[1]https://scikit-learn.org/

The GP algorithm required defining a set of functions and terminals to generate the regression models from. We defined as operators the binary operations of addition (+), subtraction (-) and multiplication (x), in addition to the analytic quotient (AQ), which has the general properties of division but without discontinuity , as proposed in [92]. The terminals are the predictive features of the datasets and ephemeral constants, which can randomly assume the values -1, 0 or 1.

After a preliminary parameter tuning, the GP was executed with an initial population of 1,000 individuals evolved by 250 generations, using a tournament selection of size 7. The probabilities of crossover and mutation were defined as 0.8 and 0.2, respectively. The depth limit of the trees was varied to show the differences in the number of nodes of the solutions found and their impact in interpretability. For the Random Forest Regressor, all the default parameters of the Sklearn tool were preserved.

In addition, we also evaluated the performance of the model with a linear regression method run with no regularization, and L1 and L2 normalization [15]. These models were chosen as they are considered interpretable regression methods (remembering that with the increasing complexity of the model, interpretability is increasingly difficult).

Tables 4.5 and 4.6 present the RMSE in the training and test sets, respectively (except for the fried and mv datasets due to the high computational cost required to run GP). Considering the non-deterministic nature of the results obtained by RF and GP, all tests presented were executed 10 times, and the median RMSE reported. Two depth of GP trees were considered: 2 and 6, to show the impact of model complexity on RMSE. The linear regression was run without any regularization and using both L1 and L2 regularization. The parameter that defines the regularization force (alpha) of the methods with L1 and L2 were chosen in order to better fit the model to the training set, performing a grid search considering values in the range from 0.1 to 6.0, in intervals of 0.3.

Note that the results of RMSE of the GP improved as we increased the depth of the tree. The results of the linear regression with and without regularization present no statistical difference for the synthetic datasets according to a t-test with 95% confidence. The results obtained with RF were better in all datasets for the training cases and overall were also the better than the GP results for the test cases. Thus, in the following experiments we will focus on the task of explaining models built by RF regression algorithms.

Table 4.5: Median RMSE in the complete training set.

Dataset	Random Forest	Genetic Programming		Linear Regression		
	10 trees	Depth 2	Depth 6	No regular.	L1	L2
keijzer-1	**0.00**	0.69	0.25	1.00	1.00	1.00
keijzer-4	**0.05**	0.99	0.45	1.00	1.00	1.00
keijzer-7	**0.00**	0.42	0.20	0.43	0.44	0.43
vladislavleva-1	**0.03**	0.71	0.31	0.77	0.79	0.77
vladislavleva-2	**0.03**	1.00	0.34	0.99	1.00	0.99
vladislavleva-3	**0.03**	1.00	0.51	1.00	1.00	1.00
vladislavleva-4	**0.13**	1.00	0.77	1.00	1.00	1.00
vladislavleva-5	**0.06**	0.47	0.07	1.00	1.00	1.00
vladislavleva-7	**0.07**	0.35	0.33	1.00	1.00	1.00
vladislavleva-8	**0.05**	0.68	0.30	0.99	1.00	0.99
cholesterol	**0.45**	0.94	0.76	0.89	0.93	0.89
auto-mpg	**0.17**	0.49	0.38	0.44	0.46	0.44
sensory	**0.38**	0.96	0.90	0.97	0.99	0.97
strike	**0.45**	0.96	0.92	0.94	0.96	0.94
day_filter	**0.15**	0.57	0.44	0.43	0.49	0.43
house	**0.18**	0.59	0.53	0.36	0.44	0.36
wineRed	**0.31**	0.86	0.81	0.78	0.82	0.78
abalone	**0.30**	0.72	0.68	0.68	0.78	0.68
wineWhite	**0.30**	0.90	0.87	0.86	0.89	0.86
cpu_act	**0.06**	0.66	0.45	0.52	0.57	0.52
bank32nh	**0.32**	0.85	0.72	0.69	0.74	0.69
puma32H	**0.12**	0.69	0.45	0.88	0.89	0.88
compactiv	**0.06**	0.63	0.35	0.51	0.56	0.51
tic	**0.52**	1.00	0.99	0.97	1.00	0.97
ailerons	**0.18**	0.58	0.49	0.42	0.48	0.42
elevators	**0.18**	0.71	0.49	0.42	0.69	0.42
california	**0.19**	0.73	0.63	0.61	0.72	0.61
house_16H	**0.27**	0.93	0.90	0.86	0.92	0.86

4.3.2 Quantitative Evaluation of DELA

This subsection evaluates the performance of the DELA method compared to the predictive capability of the local linear model approximation in relation to the original model trained with the Random Forest (RF) algorithm. It also verifies the compatibility of the feature importance returned by the explainer compared to the values returned by the model. To evaluate the local linear model, we compare the predictions using the explanation methods ELA and LIME for the same test instances. For feature importance, we compare only with ELA using all training instances. The LIME method is not used in comparing feature importance because it does not return this type of information.

Table 4.7 shows the results of the median absolute difference found when comparing the predicted values of the test points from the datasets shown in Table 4.1 for the local

Table 4.6: Median RMSE in the test set.

Dataset	Random Forest	Genetic Programming		Linear Regression		
	10 trees	Depth 2	Depth 6	No regular.	L1	L2
keijzer-1	**0.01**	0.68	0.25	1.01	1.01	1.01
keijzer-4	**0.23**	1.25	0.45	1.25	1.26	1.25
keijzer-7	**0.01**	0.39	0.16	0.39	0.39	0.39
vladislavleva-1	**0.05**	0.74	0.30	0.81	0.83	0.81
vladislavleva-2	**0.08**	0.97	0.38	1.01	1.00	1.01
vladislavleva-3	**0.05**	0.97	0.51	0.97	0.97	0.97
vladislavleva-4	**0.28**	0.98	0.77	0.99	0.98	0.99
vladislavleva-5	**0.11**	0.45	0.07	0.92	0.92	0.92
vladislavleva-7	**0.15**	0.35	0.34	0.94	0.93	0.94
vladislavleva-8	**0.08**	0.64	0.26	0.82	0.82	0.82
cholesterol	1.05	1.05	1.02	1.00	**0.94**	0.99
auto-mpg	**0.37**	0.51	0.42	0.48	0.52	0.49
sensory	**0.93**	1.04	1.01	1.04	1.04	1.04
strike	**0.89**	0.96	0.93	0.91	0.93	0.91
day_filter	**0.37**	0.59	0.48	0.49	0.51	0.48
house	**0.41**	0.61	0.56	0.56	0.53	0.56
wineRed	**0.87**	0.92	0.90	**0.87**	0.89	**0.87**
abalone	0.73	0.72	**0.70**	**0.70**	0.79	**0.70**
wineWhite	**0.78**	0.84	0.81	0.81	0.84	0.81
cpu_act	**0.15**	0.65	0.45	0.51	0.57	0.51
bank32nh	0.77	0.90	0.74	**0.71**	0.76	**0.71**
puma32H	**0.27**	0.71	0.47	0.91	0.91	0.91
compactiv	**0.14**	0.60	0.34	0.52	0.55	0.52
tic	1.10	0.95	0.95	**0.93**	0.95	**0.93**
ailerons	0.43	0.57	0.49	1.57e+12	0.47	**0.42**
elevators	0.52	0.77	0.53	**0.46**	0.76	**0.46**
california	**0.45**	0.72	0.61	0.59	0.71	0.59
house_16H	**0.62**	0.90	0.87	0.84	0.89	0.84

approximation by DELA, ELA, and LIME with the predictions returned by the RF model. The best results are highlighted in the table. Considering the non-deterministic nature of the methods involved in this process, all experiments were performed 10 times.

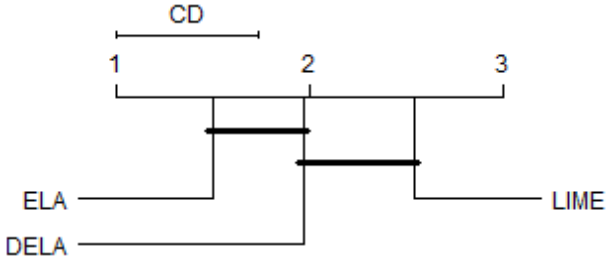
The logic behind these results is to evaluate the impact that local models have on the errors of the test points. We compare these results using an adapted Friedman test followed by a Nemenyi post-hoc test with a usual significance level of 0.05 and Bonferroni correction. The critical diagram plot is shown in Figure 4.8. As we can see, the ELA explanation method obtained a statistically superior result in the model prediction approximation compared to the values found by LIME. Regarding the DELA method, no statistical difference was found compared to the competing methods.

Since the model to be explained was trained by the RF algorithm, we can also compare the importance of the features according to the original model and compare with the global explanations of ELA and DELA. Recall that the feature importance in

Table 4.7: Median of the absolute difference found when comparing the predicted values of the test points using local approximation by ELA, DELA, and LIME with the predictions made by RF.

Dataset	ELA	DELA	LIME
keijzer-1	**0,00**	**0,00**	0,03
keijzer-4	**0,02**	**0,02**	0,27
keijzer-7	**0,00**	**0,00**	0,13
vladislavleva-1	**0,00**	**0,00**	0,08
vladislavleva-2	**0,00**	0,01	0,11
vladislavleva-3	**0,01**	0,24	0,20
vladislavleva-4	**0,04**	0,05	0,07
vladislavleva-5	**0,02**	**0,02**	0,28
vladislavleva-7	**0,14**	0,15	1,96
vladislavleva-8	**0,02**	0,03	0,76
cholesterol	20,94	22,47	**14,26**
auto-mpg	1,29	1,33	**1,23**
sensory	0,34	**0,27**	0,29
strike	100,40	**99,79**	142,73
day_filter	**353,38**	388,73	419,56
house	**13359,63**	13491,83	13541,25
wineRed	0,22	0,26	**0,21**
abalone	0,94	**0,87**	2,16
wineWhite	0,27	0,31	**0,25**
cpu_act	**1,11**	**1,11**	5,17
bank32nh	**0,03**	0,04	**0,03**
puma32H	**0,02**	**0,02**	**0,02**
compactiv	**1,10**	1,13	4,78
tic	**0,03**	0,08	0,09
ailerons	**0,00**	**0,00**	**0,00**
elevators	**0,00**	**0,00**	0,01
california	**24254,48**	25778,47	60327,90
house_16H	**7160,54**	7961,15	21560,82
fried	**1,06**	1,09	1,23
mv	0,49	**0,40**	2,88

Figure 4.8: Critical diagram comparing the output of the explanation methods with the outputs provided by the original model.



Source: created by the author.

ELA is calculated according to Equation 4.10, and in DELA, it is calculated according to Equation 4.12. These calculations are performed for each of the training instances and aggregated into output intervals as requested by the user. In the RF algorithm, the feature importance is measured by calculating the decrease in the average impurity that each attribute provides when used to construct the decision trees. By default, the Sklearn tool calculates the impurity by the Gini metric. The results obtained are presented in Table 4.8.

It is essential to observe that an RF model identifies the importance of a feature using the entire solution space of the problem. Therefore, to compare the importance of the features calculated by the RF with the explanation methods, it was necessary to delimit the global explanation generated by the model to only one output interval. Note that different regions of the solution can have different resource importance. Therefore, the ELA and DELA explanation methods have the additional advantage over the RF model of showing the importance of resources in different output intervals as needed by the user and not just in one. In addition, other models created by other regression algorithms may have no ability to estimate feature importance. In this case, using the explanation generated by explanation methods such as ELA or DELA would be the only alternative to understanding the model.

Thus, for each feature, we calculated its importance in ELA and DELA by aggregating the importances found in all training instances. We compared this result with the feature importance of the RF by calculating the RMSE between the values obtained by each of the features. In this process, we obtained as a result that there is no statistical difference between the importances calculated by the two methods during the explanation process.

These experiments show that the DELA explanation method can achieve quantitative results as good as its main competitors (ELA and LIME) during the local approximation process in obtaining explanations.

4.3.3 Qualitative Evaluation of DELA

This section presents a qualitative evaluation considering 3 datasets that have nonlinear relationships between input features and the output feature. Our interest is to verify of the results obtained by DELA make sense both in terms of the explanations for specific samples and the overall explanation provided by the method. In particular, in this section we are also interested in verifying the differences in global explanations when compared with the results obtained by ELA. The datasets analyzed were: i) wineRed; ii)

Table 4.8: RMSE of feature importance returned by the original model when compared with the ELA and DELA explanation methods.

Dataset	ELA	DELA
keijzer-1	0.00	0.00
keijzer-4	0.00	0.00
keijzer-7	0.00	0.00
vladislavleva-1	**2.77**	6.95
vladislavleva-2	0.00	0.00
vladislavleva-3	17.04	**14.26**
vladislavleva-4	**0.59**	0.60
vladislavleva-5	3.13	**2.16**
vladislavleva-7	**1.99**	2.11
vladislavleva-8	2.78	**2.29**
cholesterol	**4.76**	6.19
auto-mpg	9.31	**9.14**
sensory	**2.85**	3.93
strike	6.23	**5.93**
day_filter	10.70	**9.96**
house	6.23	**6.06**
wineRed	**6.88**	6.96
abalone	13.41	**12.91**
wineWhite	4.58	**4.54**
cpu_act	12.05	**11.96**
bank32nh	5.51	**5.37**
puma32H	**11.35**	11.36
compactiv	**14.44**	14.83
tic	**1.12**	1.22
ailerons	**6.61**	7.30
elevators	10.19	**10.54**
california	**14.71**	15.40
house_16H	**5.59**	6.90
fried	10.67	**10.61**
mv	**15.14**	15.48

auto-mpg; and iii) cholesterol.

WineRed: This dataset describes the characteristics of wines of the red type and the output the quality, with notes varying between 0 and 10. The set of 11 features used to describe each instance is as follows: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulfates, and alcohol.

We randomly selected examples of explanations to illustrate the algorithm output. Table 4.9 shows that, for a wine receiving note 3, the absolute error of the explanation compared to the model's output was 0.09 (5.20 for RF and 5.11 for DELA). In addition, looking at the importance metric, we can highlight the following features: citric acid (26.58%), alcohol (17.00%) and sulfates (14.48%).

Table 4.9: Information of the regression methods and DELA for the wineRed dataset.

Features	**Test point**	**DELA**		**Neighb. Var.**
		Feat. Import	**Coeffic**	
fixed acidity	11.6	5.13	-0.071	10.4 <= x <= 12.7
volatile acidity	0.58	2.59	-0.425	0.39 <= x <= 0.6
citric acid	0.66	26.58	0.440	0.61 <= x <= 0.66
residual sugar	2.2	2.18	0.409	2.2 <= x <= 2.6
chlorides	0.074	1.72	0.247	0.063 <= x <= 0.091
free sulfur dioxide	10.0	1.20	-0.053	6.0 <= x <= 11.0
total sulfur dioxide	47.0	11.02	0.664	18.0 <= x <= 47.0
density	1.001	12.85	0.214	0.9997 <= x <= 1.0008
pH	3.25	5.23	-0.139	3.03 <= x <= 3.25
sulfates	0.57	14.48	0.736	0.49 <= x <= 0.69
alcohol	9.0	17.0	0.595	9.0 <= x <= 9.9
Intercept DELA	-0.464			
Quality (y)	3			
DELA (y')	5.11			
RF (y')	5.2			

Finally, the last column in Table 4.9 shows the range of values of the features of neighbors considering a maximum range of 10% of variance in the output. This strategy is used by ELA to assess how different the features in the training instances selected as neighbors are. Both ELA and DELA consider that a difference in the output value between the instance of interest for explanation and the selected neighbors, exceeding 10% of the corresponding output range of all training points in the model, should be avoided. This limit is used to mitigate the risk of including noisy data in the local explanations. By verifying the modifications of the values of the features we can understand how wines similar to the presented could be manufactured. For example, the residual sugar feature could range between 2.20 and 2.60.

Since we have a large training set, we can also provide an overall explanation of the model's functioning using a graph of stacked areas[2], as shown in the Figure 4.9a. Recall that the stacked area chart is built by aggregating the local explanations of the training instances. Thus, the importance of the features is evaluated for each training instance, and then grouped into intervals according to the user's preference. In this specific case, we defined up to 9 intervals, ranging from integers 1 to 10 representing the scores assigned to wines. The graph shows that, regardless of the quality attributed to a wine, in general the most relevant features are alcohol, and features related to the amount of sulfur and acidity of wines. Figure 4.9b shows the global explanation provided by ELA, which is very similar to the one given by DELA.

[2]Generated using Chart.js[3] tool

Table 4.10: Information of the regression methods and DELA for the auto-mpg dataset - test point 1.

Features	**Test point**	**DELA**		**Neighb. Var.**
		Feat. Import	**Coeffic**	
cylinders	8	1.91	0.040	x = 8.0
displacement	429	**34.65**	-0.471	400.0 <= x <= 455.0
horsepower	208	0.27	-0.003	190.0 <= x <= 225.0
weight	4633	**24.42**	-0.382	4341.0 <= x <= 4952.0
acceleration	11	**26.98**	-0.535	10.0 <= x <= 12.5
model year	72	11.76	0.325	70.0 <= x <= 73.0
origin	1	0.0	0	x = 1.0
Intercept DELA	-0.064			
Consumption (y)	11.0			
DELA (y')	13.16			
RF (y')	12.7			

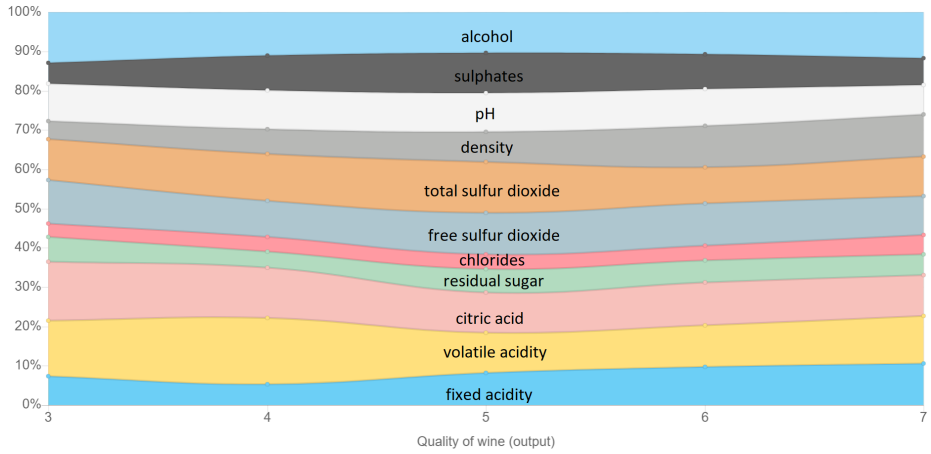
Upon comparing this explanation to the one provided by ELA in Figure 4.9c, which was obtained when the model was trained with a GP algorithm, we can observe significant changes in the importance of features. Specifically, we found that in the GP model, the alcohol and total sulfur dioxide features contributed more to the output. This indicates that both ELA and DELA methods produce consistent results when the model remains unchanged and adjust according to the model being explained, not just the initial dataset used in building the model. This type of information can be valuable for a domain expert in choosing between algorithms capable of creating the most appropriate model to their problem.

Auto-mpg: This dataset concerns city-cycle fuel consumption in miles per gallon (MPG), to be predicted in terms of 3 multivalued discrete and 5 continuous features. The set of 7 features used to describe each instance is as follows: cylinders, displacement (engine displacement), horsepower, weight, acceleration, model year, and origin.

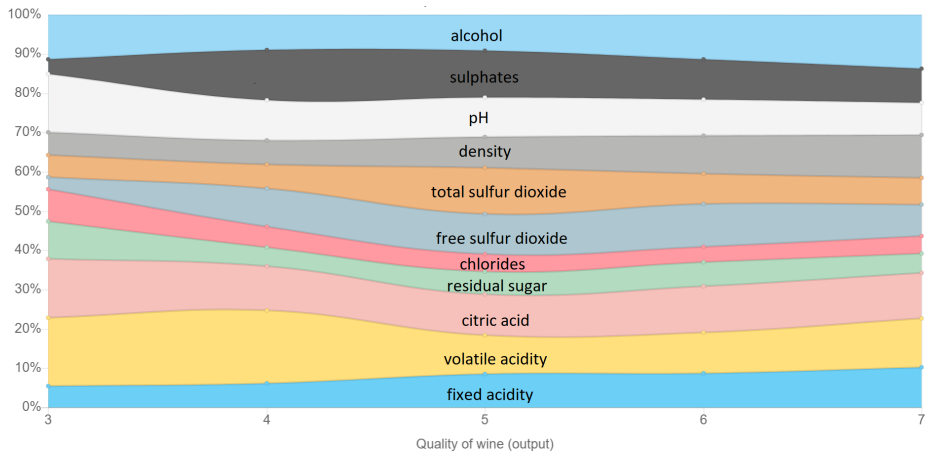
The example of explanation in Table 4.10 shows that, for a car with high fuel consumption (11 MPG), there is an absolute error of 0.46 (12.70 for RF and 13.16 for DELA). In addition, looking at the importance metric, we can highlight the following features: displacement (34.65%), acceleration (26.99) and weight (24.42%). Table 4.11 shows an example of local interpretation of a low fuel consumption car (43.1 MPG), with an absolute error of 0.89 (33.23 for RF and 34.12 for DELA). We can see that again the most important feature is weight (28.97%) followed by acceleration (28.65%) . Comparing the last lines in Tables 4.10 and 4.11, we conclude that fuel-efficient cars should be considerably lighter and do not have great acceleration power.

Finally, we provide an overall explanation of the model's functioning using a graph

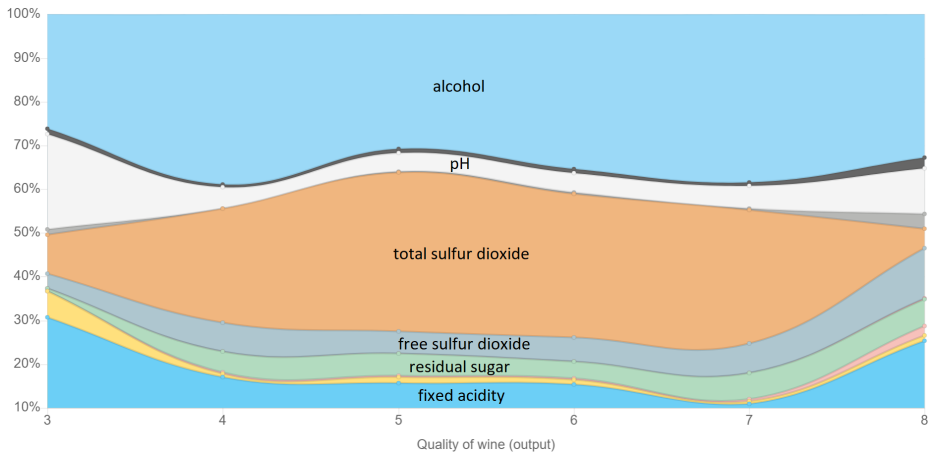
Figure 4.9: Global explanation for attributing quality to a wine. The x-axis represent the note the wine received, the y-axis the relative importance of the feature when that note is attributed to a wine.



(a) DELA - RF Model



(b) ELA - RF Model



(c) ELA - GP Model

Source: created by the author.

Table 4.11: Information of the regression methods and DELA for the auto-mpg dataset - test point 2.

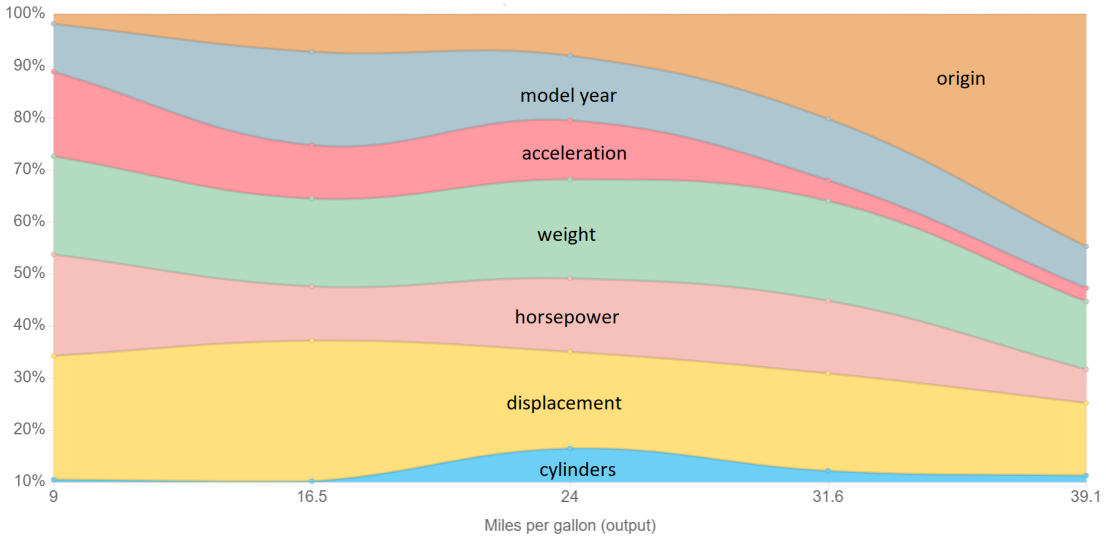
Features	**Test point**	**DELA**		**Neighb. Var.**
		Feat. Import	**Coeffic**	
cylinders	4	0.0	0	x = 4.0
displacement	90	11.84	-0.202	79.0 <= x <= 90.0
horsepower	48	18.27	-0.263	48.0 <= x <= 58.0
weight	1985	**28.97**	-0.420	1825.0 <= x <= 1985.0
acceleration	21.5	**28.65**	-0.279	18.6 <= x <= 21.5
model year	78	11.97	0.823	77.0 <= x <= 78.0
origin	2	0.29	0.137	x = 2.0
Intercept DELA	0.456			
Consumption (y)	43.1			
DELA (y')	34.12			
RF (y')	33.23			

of stacked areas, as shown in the Figure 4.10a. This show us that in general the most relevant features are displacement, weight, horsepower, and origin. According to [93] the heavier the vehicle is, the more energy it needs to get moving as heavier vehicles have greater inertia and greater rolling resistance, both contributing to increased fuel consumption. Reducing weight is a very effective way to improve a vehicle's efficiency. In addition, another parameter that influences the fuel consumption rate is the engine power [93], feature highly related to the features displacement and horsepower.

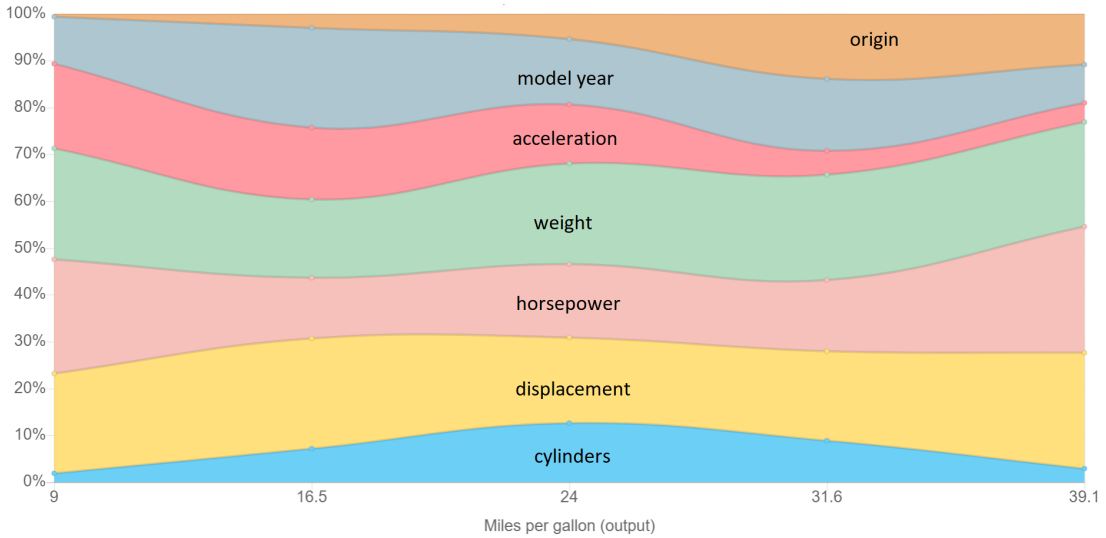
If we compare the explanations provided by DELA (Figure 4.10a) and ELA (Figure 4.10b), we can see that the importance of the origin feature differs considerably when looking at more economical cars. In order to verify which explanation is the most correct, we will resort to our dataset and verify the relationship between the origin of the car and its respective consumption. As shown in the boxplot in Figure 4.11, the American automobiles in the dataset do have a much higher fuel consumption than the others. Thus, we have a strong indication that the modifications proposed for DELA managed to obtain an even richer and more precise explanation than that obtained by ELA.

Cholesterol: This dataset describes the characteristics of individuals with the output being their blood cholesterol (mg/dl) levels. The set of features used to describe each instance is as follows: i) age in years; ii) sex; iii) cp: chest pain type (1 = typical angina; 2 = atypical angina; 3 = non-anginal pain; 4 = asymptomatic); iv) trestbps: resting blood pressure (in mm Hg on admission to the hospital); v) fbs: fasting blood sugar > 120 mg/dl (1 = true; 0 = false); vi) restecg: resting electrocardiograph results (0 = normal; 1 = having ST-T; 2 = hypertrophy); vii) thalach: maximum heart rate achieved; viii) exang: exercise induced angina (1 = yes; 0 = no); ix) oldpeak: ST depression induced by exercise

Figure 4.10: Global explanation for attributing consumption in miles per gallon. The x-axis represent the consumption, the y-axis the relative importance of the feature when that output is attributed to a car.



(a) DELA



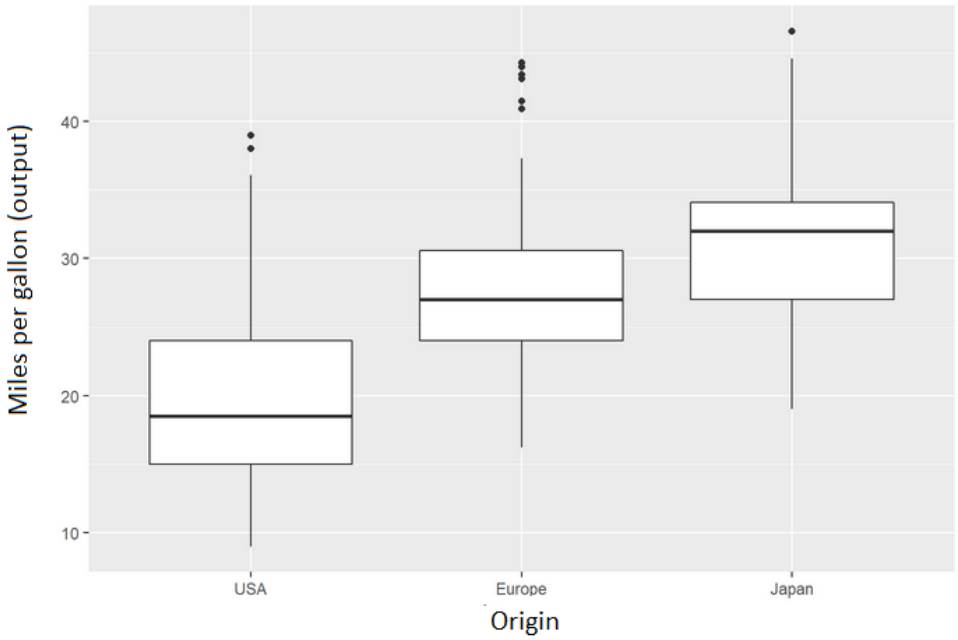
(b) ELA

Source: created by the author.

relative to rest; x) slope: the slope of the peak exercise ST segment (1 = upsloping; 2 = flat; 3 = downsloping); xi) ca: number of major vessels (0-3) colored by flourosopy; xii) thal: 3 = normal; 6 = fixed defect; 7 = reversable defect; and xiii) num: diagnosis of heart disease (angiographic disease status) (Value 0 =< 50% diameter narrowing; Value 1 => 50% diameter narrowing).

The example of explanation in Table 4.12 shows that, for a person with 335 mg/dl cholesterol, there is an absolute error of 15.34 (238.6 for RF and 253.94 for DELA). In addition, looking at the importance metric, we can see that the most important feature

Figure 4.11: Car consumption (miles per gallon) by origin.



Source: created by the author.

Table 4.12: Information of the regression methods and DELA for the cholesterol dataset.

Features	**Test point**	**DELA**		**Neighb. Var.**
		Feat. Import	**Coeffic**	
age	57	1.38	0.112	54.0 <= x <= 61.0
sex	1	0	0	x = 1.0
cp	4	0	0	x = 4.0
trestbps	110	14.10	0.789	110.0 <= x <= 152.0
fbs	0	0	0	x = 0.0
restecg	0	14.20	-0.490	0.0 <= x <= 2.0
thalach	143	**37.54**	0.788	88.0 <= x <= 143.0
exang	1	0	0	x = 1.0
oldpeak	3	15.99	-0.516	1.2 <= x <= 3.6
slope	2	0	0	x = 2.0
ca	1	0	0	x = 1.0
thal	7	0	0	x = 7.0
num	2	16.80	0.417	1.0 <= x <= 3.0
Intercept DELA	1.23			
Cholesterol level (y)	335.0			
DELA (y')	253.94			
RF (y')	238.60			

to explain the output obtained for this individual was thalach (37.53%).

We also provide an overall explanation of the model's functioning using a graph of stacked areas, as shown in the Figure 4.12a. It show that, in general, the most relevant

features are *sex*, *ca*, *thalach*, and *age*.

Researchers already know that there is a relationship between sex, age [58] and an individual's cholesterol level. It is known that from the age of 20-55, men tend to have higher cholesterol levels, but after that age, cholesterol levels rise rapidly in women and exceed those in men [50]. As for the heartbeat frequency, it is known that when the LDL level of cholesterol gets too high a plaque can build up along the blood vessel walls, which can cause the blood vessels to become hard and narrow. This hardening of the blood vessels can restrict blood flow to the heart and or brain. When less blood is able to get through the blood vessels, the heart must beat faster to deliver enough blood and oxygen. This can lead to a higher pulse and higher blood pressure [91].

Finally, if we compare the explanations provided by DELA (Figure 4.12a) and ELA (Figure 4.12b), we observe that DELA is able to maintain a more constant behavior, without abrupt changes in terms of the importance of the features with the increase in the cholesterol level. We believe that more drastic modifications occur in ELA due to the need for instances to always be explained by only 5 neighbors. Therefore, when there is a higher local density, DELA is able to stand out by creating a more robust local model.

4.3.4 A Case Study of DELA for a medical dataset

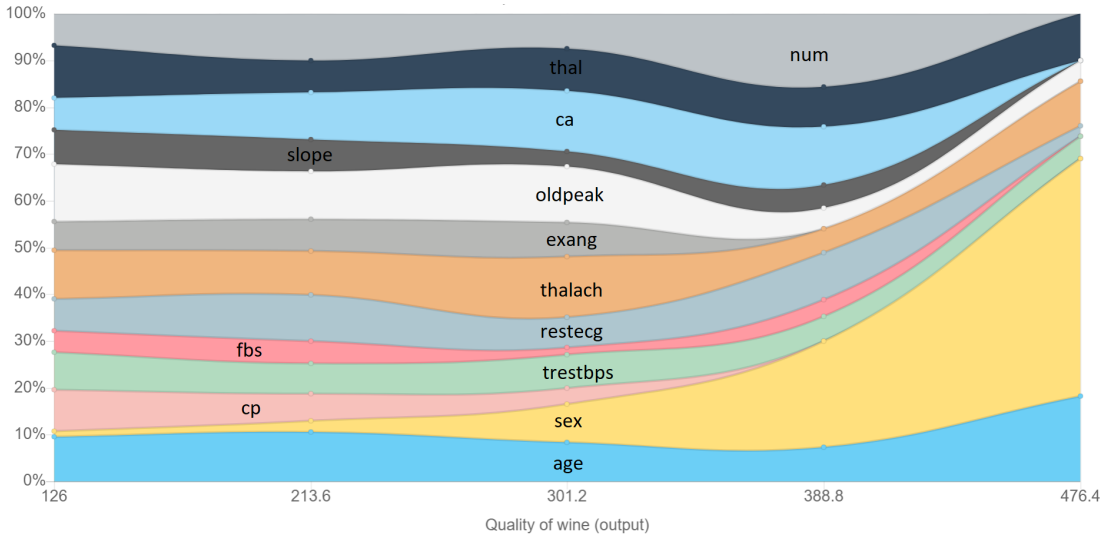
In the previous sections we analyzed the results of DELA in classical datasets from the literature, and looked at the quality of the interpretations from a layman point of view. In this section, we apply DELA to a medical dataset that aims to identify whether a patient has Rheumatic Heart Disease (RHD), and for which we have specialists that can validate the results of explanation.

RHD is a "damage to one or more heart valves that remains after an episode of acute rheumatic fever (ARF) is resolved. It is caused by an episode or recurrent episodes of ARF, where the heart has become inflamed. The heart valves can remain stretched and/or scarred, and normal blood flow through damaged valves is interrupted. Blood may flow backward through stretched valves that do not close properly, or may be blocked due to scarred valves not opening properly. When the heart is damaged in this way, the heart valves are unable to function adequately, and heart surgery may be required"[4].

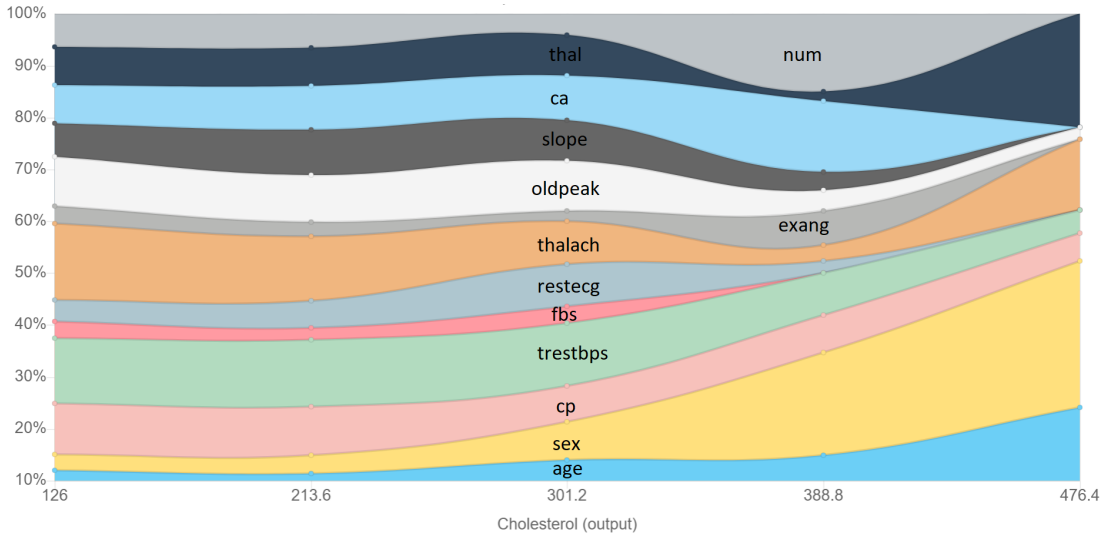
This dataset has 2,619 patients and initially targets the patient's classification in three possible classes: normal, borderline RHD, and definite RHD. As our tool was developed for regression problems we turn the problem into detecting a level of severity of the diagnosis produced: normal as level 1, borderline as level 2 and definite as level

[4]https://www.rhdaustralia.org.au/what-rheumatic-heart-disease

Figure 4.12: Global explanation for attributing cholesterol. The x-axis represent the serum cholesterol in mg/dl, the y-axis the relative importance of the feature when that output is attributed to a person.



(a) DELA



(b) ELA

Source: created by the author.

3. For each patient, 20 features are listed, as described in Table 4.13. It's important to point out that, in this dataset, 15 out of 20 features are categorical, and all of them are binary.

In this section we want to qualitatively assess the explanations obtained for different levels of disease severity. First, we randomly selected a test patient for each severity level and all other instances of the dataset were used for training. The results of this local explanation are shown in Section 4.3.4.1. Next, we aggregate all feature importance results obtained individually for each of the severity levels present in the complete training

Table 4.13: RHD: features used to describe instances.

Id	Feature	Type
x_1	Gender	Categorical
x_2	Type of Household	Categorical
x_3	# House Residents	Discrete
x_4	# House Residents Younger than 15 Years Old	Discrete
x_5	Anterior Mitral Leaflet Thickness	Discrete
x_6	Mitral Regurgitation	Categorical
x_7	Length of Mitral Regurgitation	Discrete
x_8	Mitral Regurgitation > 1.5cm	Categorical
x_9	Mitral Regurgitation > 2cm	Categorical
x_{10}	Pan-systolic by color Doppler	Categorical
x_{11}	Pan-systolic by spectral Doppler	Categorical
x_{12}	Mitral Velocity > 3m/sec (spectral Doppler)	Categorical
x_{13}	Mitral Stenosis	Categorical
x_{14}	Aortic Regurgitation	Categorical
x_{15}	Aortic Regurgitation > 1cm	Categorical
x_{16}	Length of AR Jet	Discrete
x_{17}	Pan-diastolic by color Doppler	Categorical
x_{18}	Pan-diastolic by spectral Doppler	Categorical
x_{19}	Aortic Velocity > 3m/sec (spectral Doppler)	Categorical
x_{20}	Aortic Stenosis	Categorical

dataset and plot the stacked areas graph depicting the evolution of feature importance with the addition of gravity of the disease (Section 4.3.4.2). In addition, we compare the explanations of the proposed method with the explanations obtained by LIME.

4.3.4.1 Local interpretation

In this section we show the results found by DELA for three patients, each with a different level of severity of the disease. For each patient, we report the results of the RF and DELA. Furthermore, we compare the explanations obtained by DELA with explanations provided by two other methods: ELA and LIME [107]. We will work with LIME without normalizing the input data due to the difficulty in going back to the original attributes values after normalization when presenting the results. We believe that LIME will not be harmed, since, as we previously evaluated, it is not very sensitive to the data normalization process.

Patient 1: normal RHD As shown in Table 4.14, the predicted value found by the

Table 4.14: Local explanation for diagnosis of RHD in patient level 1: normal.

Patient level 1 Test point analyzed = [0, 1, 14, 2, 27, 1, 7, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0] **Value predicted by RF:** 1.1		
DELA	**ELA**	**LIME**
Value predicted by Local Explanation:		
1.0	1.0	0.99
Coefficients of linear regression:		
[0.04, 0.01, 0.00, 0.00, 0.00, -0.06, 0.00, 0.00, 0.00, 0.04, -0.01, 0.00, 0.00, 0.00, -0.01, 0.00, -0.01, 0.00, 0.00, 0.00]	[0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00]	Not applicable
Importance of features:		
26.71, 1.84, 0.00, 0.00, 0.00, 37.10, 0.00, 0.00, 0.00, 32.68, 0.46, 0.00, 0.01, 0.00, 0.49, 0.00, 0.61, 0.05, 0.034, 0.01	0.00, 0.00, 39.11, 15.10, 27.78, 0.00, 18.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00	Not applicable
Intercept:		
-0.235	-0.271	1.510

Figure 4.13: Coefficients returned for local explanation via LIME method.

(a) Patient 1 (b) Patient 2 (c) Patient 3

Source: created by the author.

local explanation obtained by DELA (1.0) and ELA (1.0) was closer to that found by the RF (1.1) when compared with LIME (0.99).

Also shown in Table 4.14, an advantage that we can highlight at this point is that DELA and ELA provides an additional interpretability feature that is the importance of features for regression of the test instance, which does not occur in LIME. For this particular patient, we can see that DELA and ELA differ in terms of the importance of the features. DELA considers the most important feature as x_6 (Mitral Regurgitation) while ELA given a lot of importance to x_3 (# House Residents).

Note that with LIME, we do not have direct access to the coefficients used in the local linear regression. Additionally, for the ELA method, the coefficients of the linear regression were all zero. This may have happened because the sample size provided by the neighborhood was very small, with only the 5 closest instances selected. Therefore, for this specific patient, the linear model used to obtain the explanation did not have enough information to find a significant relationship between the independent and dependent variables.

Table 4.15 displays the variations in features values that identify which characteristics are sufficient for a patient to be considered locally close, i.e., patients who also have a similar severity diagnosis. The table demonstrates that the tools have different approaches for this step. DELA evaluates neighbors with a maximum of 10% variance in attributes, whereas LIME perturbs the test instance.

As a result, the outcomes obtained by LIME and DELA differ significantly. For instance, LIME indicates that patients who are in close proximity to Patient 1 have a "Length of AR Jet" (x_{16}) feature equal to or less than 0.00. However, this feature cannot take negative values since it is a measure of length, which implies that this interpretation disregards the semantic assumptions of the target problem. In other words, the perturbation process resulted in the creation of semantically invalid instances that were considered in the problem. This issue does not arise in DELA.

Furthermore, in LIME local explanations can be visualized by observing the local contribution of each feature. Figure 4.13a reports the local explanations provided by LIME for the normal RHD patient. The blue bars represent negative contributions and the orange bars contributions to increase the output of the specific test instance feature. Thus, LIME states that $Length\ of\ AR\ Jet$ has the highest local contribution. In this case, when $Length\ of\ AR\ Jet \leq 0$, its local negative contribution will be -0.34.

In general, we believe that the explanation methods DELA and LIME should not be seen as competitors, as each of them explains the model taking into account a distinct aspect. We argue that these methods can be used together so that the user has more elements to understand the output obtained by the model for a particular instance.

Patient 2: borderline RHD Table 4.16 and Figure 4.13b show the results obtained for a patient with level 2 severity. As can be seen, the result obtained by the local approximation resulting from the process performed by DELA (1.82) is closer to the

Table 4.15: Patient level 1: local variation in features that would allow finding patients with similar severity levels.

DELA	**ELA**	**LIME**
Neighbors within a maximum range of 10% of variance		Test instance perturb
$x_1 = 0.0$	$x_1 = 0.0$	$x_{16} <= 0.00$
$x_2 = 1.0$	$x_2 = 1.0$	$x_9 <= 0.00$
$12.0 <= x_3 <= 14.0$	$12.0 <= x_3 <= 14.0$	$0.00 < x_7 <= 8.00$
$2.0 <= x_4 <= 4.0$	$2.0 <= x_4 <= 4.0$	$x_{13} <= 0.00$
$13.0 <= x_5 <= 36.0$	$19.0 <= x_5 <= 31.0$	$x_{19} <= 0.00$
$x_6 = 1.0$	$x_6 = 1.0$	$x_{18} <= 0.00$
$4.0 <= x_7 <= 12.0$	$4.0 <= x_7 <= 10.0$	$x_{14} <= 0.00$
$x_8 = 0.0$	$x_8 = 0.0$	$x_{12} <= 0.00$
$x_9 = 0.0$	$x_9 = 0.0$	$x_3 > 10.00$
$x_{10} = 1.0$	$x_{10} = 1.0$	$x_{17} <= 0.00$
$x_{11} = 0.0$	$x_{11} = 0.0$	$0.00 < x_{10} <= 1.00$
$x_{12} = 0.0$	$x_{12} = 0.0$	$x_{20} <= 0.00$
$x_{13} = 0.0$	$x_{13} = 0.0$	$20.00 < x_5 <= 27.00$
$x_{14} = 0.0$	$x_{14} = 0.0$	$x_{15} <= 0.00$
$x_{15} = 0.0$	$x_{15} = 0.0$	$x_8 <= 0.00$
$x_{16} = 0.0$	$x_{16} = 0.0$	$x_{11} <= 0.00$
$x_{17} = 0.0$	$x_{17} = 0.0$	$x_4 <= 2.00$
$x_{18} = 0.0$	$x_{18} = 0.0$	$x_1 <= 0.00$
$x_{19} = 0.0$	$x_{19} = 0.0$	$0.00 < x_6 <= 1.00$
$x_{20} = 0.0$	$x_{20} = 0.0$	$x_2 <= 1.00$

RF (1.3) when compared to the value predicted by ELA (2.06). However, the local approximation generated by LIME was closest to the model output (1.36).

Regarding the importance of features for the regression performed in the test sample, DELA considers "Mitral Regurgitation > 2cm"(x_9) as the most relevant while ELA accounts for "Length of Mitral Regurgitation" (x_7), both correlated features. In the explanation provided by LIME, we can observe that locally the two features that contribute the most to the patient with severity level 2 are precisely "Length of Mitral Regurgitation" (x_7) and "Mitral Regurgitation > 2cm" (x_9), with the former having a negative value of -0.32 and the latter having a positive value of 0.21. According to the clinicians, these are the two most relevant features in assessing RHD when looking at echocardiograms.

Again, in the process of interpreting the variation of features, Table 4.17 shows that LIME creates and considers semantically invalid instances, which may lead to a misinterpretation of the evaluated problem.

Furthermore, note that ELA could not find training neighbors within the established limit, while the flexibility of DELA allowed it to find neighbors and report features variations within the severity level would remain similar. We believe this difference is mainly due to the more appropriate similarity measure to this specific dataset than the

Table 4.16: Local explanation for diagnosis of RHD in patient level 2: borderline RHD.

Patient level 2		
Test point analyzed = [1, 1, 12, 3, 13, 1, 146, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]		
Value predicted by RF: 1.3		
DELA	**ELA**	**LIME**
Value predicted by Local Explanation:		
1.82	2.06	1.36
Coefficients of linear regression:		
[-2.19e-01, -4.27e+08, 6.01e-01, 2.23e-01, -4.84e-01, 4.35e+10, 2.88, -4.39e+10, -1.75e+11, -3.34e-02, 2.742e+09, 0.00, -6.86e+08, -5.495e+09, 5.49e+09, 0.00, 5.49e+09, 0, 1.37e+09, 0.00]	[0.58, -0.03, -0.03, 1.18, -3.03, -0.09, 3.98, 0.00, 0.37, -2.03, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00]	Not applicable
Importance of features:		
0.00, 0.02, 0.00, 0.00, 0.00, 5.11, 0.00, 11.68, 82.76, 0.00, 0.05, 0.00, 0.00, 0.14, 0.10, 0.00, 0.13, 0.00, 0.01, 0.00	2.75, 0.05, 0.18, 0.72, 12.09, 0.28, 71.21, 0.00, 4.51, 8.21, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00	Not applicable
Intercept:		
6.17e+11	-18.57	1.28

generic use of a Euclidean measure. For this specific dataset, the distance measure selection heuristic proposed for DELA chose the Chebyshev distance. In addition, by the local density criterion, the DELA method selected 9 nearest neighbors of the evaluated instance for local explanation creation. Recall that in ELA the number of neighbors was fixed at 5.

Patient 3: definite RHD The local interpretation for a patient with severity level 3 is shown in Table 4.18 and Figure 4.13c. Again, the local output from DELA (2.01) was closer to the original RF model (2.5) than the results of LIME (1.37) and ELA (7.83). Again, the most important features found by DELA were "Mitral Regurgitation $>$ 2cm"(x_9) while ELA considers as the most relevant "Length of Mitral Regurgitation" (x_7), both correlated features. Furthermore, the explanation provided by LIME also indicates that the greatest local contributions to achieving the result came from the features "Length of Mitral Regurgitation" (x_7), with a negative value of -0.29, and "Length of Mitral Regurgitation" (x_9), with a positive value of 0.20. We consider this as a strong indication of consistency in the explanation given.

Finally, Table 4.19 shows that LIME produced instances with semantically invalid feature values in the test case perturbation process and ELA found no neighbors within the allowed limit.

Table 4.17: Patient level 2: local variation in features that allow finding patients with similar severity levels.

DELA	**ELA**	**LIME**
Neighbors within a maximum range of 10% of variance		Test instance perturb
$0.0 <= x_1 <= 1.0$		$x_{16} <= 0.00$
$x_2 = 1.0$		$x_9 > 0.00$
$9.0 <= x_3 <= 12.0$		$x_7 > 13.00$
$1.0 <= x_4 <= 3.0$		$x_{19} <= 0.00$
$10.0 <= x_5 <= 13.0$		$x_{11} <= 0.00$
$x_6 = 1.0$		$x_{18} <= 0.00$
$146.0 <= x_7 <= 153.0$		$x_5 <= 15.00$
$x_8 = 1.0$		$x_{13} <= 0.00$
$x_9 = 1.0$		$x_3 > 10.00$
$0.0 <= x_{10} <= 1.0$	Did not find neighbors.	$x_{17} <= 0.00$
$x_{11} = 0.0$		$0.00 < x_{10} <= 1.00$
$x_{12} = 0.0$		$2.00 < x_4 <= 3.00$
$x_{13} = 0.0$		$x_{15} <= 0.00$
$x_{14} = 0.0$		$x_{12} <= 0.00$
$x_{15} = 0.0$		$x_{20} <= 0.00$
$x_{16} = 0.0$		$0.00 < x_6 <= 1.00$
$x_{17} = 0.0$		$0.00 < x_1 <= 1.00$
$x_{18} = 0.0$		$x_8 > 0.00$
$x_{19} = 0.0$		$x_{14} <= 0.00$
$x_{20} = 0.0$		$x_2 <= 1.00$

4.3.4.2 Global interpretation

As mentioned earlier, DELA and ELA can also provide a higher and more global level of interpretation using a stacked area chart. As shown in Figures 4.14a and 4.14b, the explanations provided by both DELA and ELA indicate, from level 1 patients (normal RHD), that mitral valve-related features play a main role in diagnosis. In addition, we can observe that characteristics related to the socioeconomic situation of the patients also exert considerable influence on the prediction of the diagnosis, such as the number of residents and the type of house. By increasing the critical level, we observed that the features related to the thickness of the mitral valve become even more relevance.

These results are consistent with those used in clinical practice. RHD diagnosis is mainly based on the thickness of the Mitral Valve, the length of regurgitation jet and socioeconomic information, since RHD usually affects more people in lower social classes. These results validate the proposed explanation method in a real case.

Table 4.18: Local explanation for diagnosis of RHD in patient level 3: definite RHD.

Patient level 3 Test point analyzed = [0, 1, 9, 3, 43, 1, 25, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] **Value predicted by RF:** 2.5		
DELA	**ELA**	**LIME**
Value predicted by Local Explanation: 2.01	7.83	1.37
Coefficients of linear regression: [-8.69e+09, -2.52e+11, -5.42e-01, -3.31e-01, -2.91e+00, 6.54e+11, -7.07e-01, 0.00, -2.61e+12, 0.00, 0.00, 0.00, -2.045e+10, -1.63e+11, 1.63e+11, 0.00, 1.63e+11, 0.00, 4.09e+10, 0.00]	[0.00, 0.00, -1.703, 17.539, 0.757, 0, 82.123, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00]	Not applicable
Importance of features: 0.075, 1.07, 0.0, 0.0, 0.0, 5.75, 0.0, 0.0, 92.25, 0.0, 0.0, 0.0, 0.01, 0.31, 0.23, 0.0, 0.28, 0.0, 0.01, 0.0	0.0, 0.0, 1.46, 8.09, 5.79, 0.0, 84.66, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0	Not applicable
Intercept: 8.07e+12	-12.75	1.34

4.4 Summary

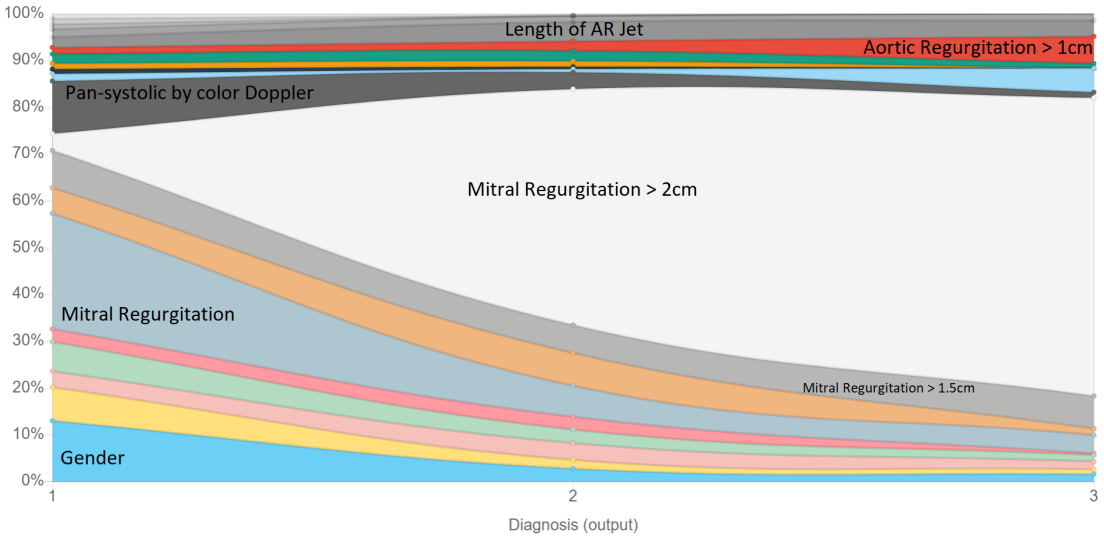
This chapter focused on improving ELA to encompass a more dynamic internal structure that is adaptable to the characteristics of each dataset, named DELA (Dynamic Explanation by Local Approximation). DELA chooses the most appropriate distance measure to the dataset at hand, dynamically defines the number of neighbors that will be selected for the local explanation according to the data local density and calculates the importance of the features based on the location of the test instance.

As a result, DELA is able to obtain local explanations with errors that are, overall, statically similar to ELA, with the advantage of being less affected by datasets with high variance in features. Furthermore, qualitatively DELA was able to identify the importance of features that were underestimated by ELA and find a greater number of neighbors in denser regions to provide more robust local explanations. Compared to LIME, DELA was able to return semantically valid local patterns in its explanations, something that does not always occur with its competitor.

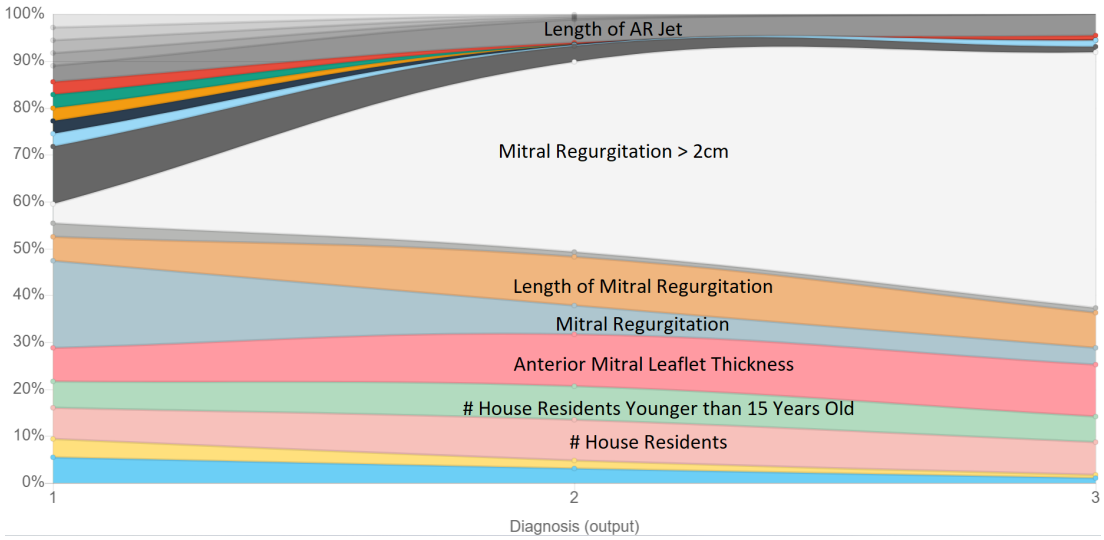
Table 4.19: Patient level 3: local variation in features that would allow finding patients with similar severity levels.

DELA	**ELA**	**LIME**
Neighbors within a maximum range of 10% of variance		Test instance perturb
$x_1 = 0.0$		$x_{16} <= 0.00$
$x_2 = 1.0$		$x_9 > 0.00$
$9.0 <= x_3 <= 10.0$		$x_7 > 13.00$
$x_4 = 3.0$		$x_{18} <= 0.00$
$x_5 = 43.0$		$x_{12} <= 0.00$
$x_6 = 1.0$		$x_5 > 27.00$
$19.0 <= x_7 <= 25.0$		$x_{17} <= 0.00$
$x_8 = 1.0$		$x_{13} <= 0.00$
$x_9 = 1.0$		$x_{19} <= 0.00$
$x_{10} = 1.0$	Did not find neighbors.	$x_{14} <= 0.00$
$x_{11} = 0.0$		$x_{11} <= 0.00$
$x_{12} = 0.0$		$x_{20} <= 0.00$
$x_{13} = 0.0$		$0.00 < x_{10} <= 1.00$
$x_{14} = 0.0$		$x_{15} <= 0.00$
$x_{15} = 0.0$		$x_3 <= 9.00$
$x_{16} = 0.0$		$x_1 <= 0.00$
$x_{17} = 0.0$		$x_8 > 0.00$
$x_{18} = 0.0$		$0.00 < x_6 <= 1.00$
$x_{19} = 0.0$		$2.00 < x_4 <= 3.00$
$x_{20} = 0.0$		$x_2 <= 1.00$

Figure 4.14: Global explanation for attributing diagnosis of RHD. The x-axis represent the diagnosis, the y-axis the relative importance of the feature when that output is attributed to a patient.



(a) DELA



(b) ELA

Source: created by the author.

Chapter 5

M-PEER: An example-based explanation method

In this chapter we present our contribution to example-based/prototype-based explanation methods. In this type of approach we look for or create instances that can represent the black-box model or a specific test instance prediction. We call these explanatory instances *prototypes*.

The set of selected prototypes needs to be able to both describe the data distribution considering the fitted machine learning model while also explaining individual predictions. We propose a new methodology that divides the explanation process into two distinct steps, and is able to provide both global and local explanations. Global explanations refer to understanding the model's overall behavior: *What are the patterns underlying the model's behavior?* Alternatively, local explanations are in the context of an individual prediction: *Which points in the training set most closely resemble a test point?* In our methodology, the first step refers to identifying a set of data representative prototypes. Next, we locally explain each prediction by assigning it to its closest prototype, according to a metric proposed in this work.

We assume that a prototype refers to an instance in the training set, and the set of prototypes is representative of the whole dataset. To find the prototypes, we propose M-PEER (Multiobjective Prototype-basEd Explanation for Regression), a method based on Pareto multi-objective evolutionary optimization. M-PEER extracts post-hoc explanations by treating the original model as a black-box [105], and hence is model-agnostic. It requires as input the predictions a regressor fitted using a dataset, and returns a set of prototypes that globally explains the model.

M-PEER was conceived to optimize three metrics: global prototypes fidelity, global prototypes stability, and the Root Mean Squared Error (RMSE). The first two metrics were inspired by neighborhood-fidelity (NF) [98] and stability (S) [83] metrics previously proposed in the literature of interpretability for classification. While these two metrics were originally proposed to capture the quality of local explanations, we extrapolate them to also evaluate the model global explanation.

The global fidelity measure wants the chosen prototypes to be as close as possible

to the output of the instances to be explained. In contrast with previous approaches, we account for the model's output when calculating prototypes fidelity and not only the input data. We believe that without considering the model output the measure simply explains the data and not the model learned by the regression algorithm. The second measure, global stability, wants the features of the prototypes to be as similar as possible to the features of the instances to be explained. Finally, we need to account for the error, as we want the error generated by the prototypes to be as close as possible to the error of the black-box model. As we do not have a clear and quantitative way to define which of these objectives is the most relevant, the Pareto multi-objective approach is a good fit to this problem.

We compare the results of our prototype-based method with ProtoDash [51], the state-of-the-art method in the context of explanation via prototypes. We also present comparisons to other machine learning methods that could be easily adapted to select prototypes: clustering, hub-based approaches and instance selection. Results demonstrate significant gains of M-PEER over other strategies, with an average of 12% improvement in the proposed Global Fidelity and Stability (GFS) metric and 17% in RMSE when compared to ProtoDash. In particular, the M-PEER version that optimizes three objectives (i.e., RMSE, fidelity and stability) was always superior than the state-of-the-art in all GFS measures, and up to 80% better in terms of RMSE. The main contributions of this work are:

1. To the best of our knowledge, this is the first work dedicated specifically to finding and evaluating prototypes in order to obtain explanations for models in regression problems, taking into account characteristics such as the output of the regressor model and not only the features of the instances.

2. We introduce a new two-level methodology to choose prototypes from training data, capable of providing both global and local explanations to predictions.

3. We propose a multi-objective evolutionary method to find the best prototypes for explaining the black-box model, called M-PEER.

4. We adapt and empirically show the applicability of strategies already used in other areas of machine learning in order to select the best prototypes to explain the model built by regressors, and show the proposed method outperforms them.

Algorithm 4 Prototype-based methodology to explain regression models.

Require: T (train), T' (test), e (number of prototypes)
1: f = Regression model(T)
2: PR = Select e prototypes(T, f)
3: Show global model explanation(PR)
4: **for** each $t' \in T'$ **do**
5: $f(t') = \hat{y}$
6: $pr = \arg\min_{pr} \|dist(t', PR_{pr})\|$
7: Show prototype pr as a local explanation for $f(t')$
8: **end for**

5.1 Proposed Methodology

This section introduces the methodology proposed to provide both global and local explanations to regression models. Given a finite set of input-output pairs representing the training instances $T = \{t_i = (\mathbf{x}_i, y_i)\}_{i=1}^{n}$—with $(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$ and $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{id}]$, for $i = \{1, 2, \ldots, n\}$—we define $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^T$ and $Y = [y_1, y_2, \ldots, y_n]^T$ as the matrix $n \times d$ of inputs and the n-element output vector, respectively. We then formulate a regression problem as the problem of learning a function $\hat{f}_\beta : \mathbf{x}_i \rightarrow \mathbb{R}$ such that $\hat{f}(\mathbf{x}_i; \beta) \simeq f(\mathbf{x}_i) = y_i, \forall \mathbf{x}_i \in X, \forall y_i \in Y$, where $f(\mathbf{x}_i)$ stands for the true unknown function that generates the samples output values, and $\hat{f}(\mathbf{x}_i; \beta) = \hat{y}_i$ stands for an approximation dependent on the feature vector $\mathbf{x}_i$, and an unknown parameter vector $\beta \in \mathbb{R}^n$.

Global explanations refer to understanding the model's overall behavior, and are given by a set of prototypes PR selected from T. The local explanations, in turn, are in the context of individual predictions. Given a test point t', the explanation is obtained by assigning t' to its closest prototype in PR. Recall that the methodology is agnostic in relation to the regression model.

Algorithm 4 illustrates the proposed methodology. It starts by learning a regression model f, and then selects e prototypes PR from the training set (line 2). These selected prototypes are responsible for the global explanation of the model (line 3). Next, for each test point t', we associate it to its closest prototype in PR according to a distance function (line 6), allowing for a local explanation (line 7). Next we present a method we propose for instantiating the aforementioned approach – which selects the most suitable prototypes to explain regression models (Section 5.1.1) and then assigns the selected prototypes to test examples (Section 5.1.2)– called M-PEER.

Algorithm 5 Evolutionary process - M-PEER

Require: N (population size), P_{xover} (probability of crossover), P_{mut} (probability of mutation), G (number of generations)

1: Pop = Generate initial population
2: $A = \emptyset$
3: **while** $current\_g <= G$ **do**
4: Fit = Evaluate(Pop, A)
5: A = Selection($Pop \cup A$, N)
6: **if** $current\_g <= G$ **then**
7: Pop = inds for next generation (A, P_{xover}, P_{mut})
8: **end if**
9: **end while**
10: A^* = Pareto non-dominated solutions(A)
11: $Bestind = \arg\min_i \|GFS(A_i^*)\|$
12: **return** $Bestind$

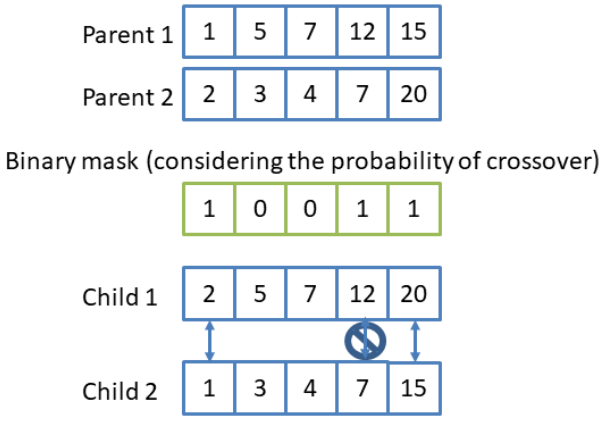
5.1.1 Strategy for Selecting Prototypes

This section presents the method we propose to select prototypes suitable for explaining regression model predictions. As detailed in Alg. 4, the method returns the model's global explanation, represented by a set of prototypes PR and, for each test instance, a local explanation. Recall that the local models are explained by a single prototype borrowed from the global model. Hence, in order to select the prototypes of the global model we also need to account for their local quality. With that in mind, we search for a set of prototypes that minimizes errors while also improving the metrics of global fidelity (GF) and global stability (GS), as detailed later in this section.

To accomplish this task we model a multi-objective evolutionary algorithm to the problem, more specifically SPEA2 (Strength-Pareto Evolutionary Algorithm) [124]. SPEA2 follows the same principles of other evolutionary algorithms. Furthermore, SPEA2 evaluates solutions using more than one criterion, and decides which solutions are the best based on the concepts of Pareto dominance. The concept of Pareto dominance is appropriate for cases where we have metrics that are conflicting but equally important to solve the problem. According to this concept, a solution s_1 dominates a solution s_2 if it is better than s_2 in at least one criterion and not worse in any other criteria considered when solving the problem.

Algorithm 5 illustrates M-PEER. Each individual represents a solution to the problem, i.e., a set of prototypes encoded as a list of e positions– where e is the number of desired prototypes and a prototype is represented by a training example. When the algorithm starts, a population Pop is generated by randomly assigning unique training instances to the e positions of the vector (line 1). As M-PEER is based on SPEA2, it also

Figure 5.1: Example of crossing over two individuals in GA. In this case, each individual stores 5 prototypes. Note that elements 12 and 7 are not exchanged, as there would be a repetition of one prototype (number 7) in one of the offspring individuals.



Source: created by the author.

works with an archive A, which stores the best solutions found by the algorithm so far (line 2). The population evolves until a stop criterion is met (line 3) – which in our case was defined as a limit number of generations G, where evolution consists of individuals being selected to undergo crossover and mutation operators according to their quality.

The quality of an individual is measured by a fitness function (line 4), which depends on how good the selected prototypes are at explaining a model's global and local predictions, and is measured by three different objectives/measures, detailed later in this section. The fitness of each individual i depends on the number of solutions in Pop and A that dominate i according to the Pareto dominance criterion, and the density of the region where a solution is – this last criteria ensures population diversity, a desirable property in multi-objective optimization.

After solutions are evaluated, the archive A is updated to store the best individuals in Pop and A (line 5). Initially, all non-dominated individuals are selected to be in A. If there is still free space in A (i.e., size of $A \leq N$), the best individuals among the dominated will be added according to the fitness function until A is full. If the size of $A \geq N$, a truncation is performed by iteratively removing individuals from denser regions.

After this, individuals are probabilistically selected to undergo crossover and mutation operations, generating a new population (line 7). We use the tournament selection scheme, where l individuals are randomly selected from the population and the one with the best fitness value is selected. Pairs of selected individuals may undergo uniform crossover according to a probability P_{xover}, where each position of the selected individuals are probabilistically exchanged, as exemplified in Figure 5.1). Mutation works in a similar fashion according to a probability P_{mut}, but involves a single individual, where a selected gene is replaced by a random example to represent the prototype. Both operations guarantee the generated individuals are composed of unique prototypes, and are

applied according to user defined probabilities.

After the stopping criterion is satisfied, the non-dominated individuals are selected to compose the solution set A^* (line 10). The best individual, returned as the solution to our problem (line 12), is defined as that among the non-dominated set of solutions A^* that minimizes GFS (Eq. 5.7) when assigning prototypes, as explained next.

Fitness objectives: Defining the quality of an explanation model, which here corresponds to the objectives of our optimization problem, is not a trivial task. During the post-hoc explanation process, an alternative model is used to locally replace the more complex black-box model. Usually this model provides an approximation to the original model, i.e., the explanation strategy generates an error when compared to the original model. Hence, our first objective measures this error using the Root Mean Squared Error ($RMSE$), shown in Eq. 5.1.

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{i=1}^{|T|}(y_i - \hat{y}_i))^2} \tag{5.1}$$

where $|T|$ represents the size of the training set, $\hat{y}_i$ and y_i denote the output of the training instances when the model is created using only the selected prototypes and when the complete original model is used, respectively.

Furthermore, we also introduce to the fitness what we call “semantic objectives", which are ways to measure if the explanation will facilitate human understanding and raise the confidence in the results. Inspired by two metrics previously proposed in the literature of explainability for classification, we adapt the stability (S) [83] and neighborhood-fidelity (NF) [98] metrics, generating what we call global stability (GS) and global fidelity (GF). While the original S and NF metrics were developed to capture the quality of local explanations, we use an aggregated version of them to evaluate the model global explanation, as detailed next. These two metrics will also be optimized together with the error.

The idea of local explanations is, given an instance t' to be explained, to approximate the black-box model f across some neighborhood $N_{t'}$ using a function g. Neighborhood-fidelity (NF) is defined in [98] as the squared difference between the values predicted by the local approximation function g (explanation) and the values predicted by the black-box model f. As our method is based on prototypes, we define the neighborhood $N_{t'}$ around an instance t' to be of size one. This means that we select the prototype PR that best explains the instance of interest. Hence, given the instance t' to be explained, we define the Local Fidelity (LF) as the square of the difference between the output y_i of the black-box model f for t' and the output y_j of the prototype PR selected from the training set to explain t' (PR(t')), as shown in Eq. 5.2.

$$LF(f(t'), PR(t')) = dist(y_i, y_j) = (y_i - y_j)^2 \tag{5.2}$$

Based on that, we define the Global Fidelity (GF) as the median of the Local Fidelity (LF) for all instances to be explained, as shown in Eq. 5.3. When calculating the fitness, we explain the instances in the training set.

$$GF(f, PR_t, t) = median_{\forall t \in T}(LF(f(t), PR(t))) \tag{5.3}$$

The second measure adapted was stability (S), proposed in [83] to capture whether the explanations are robust with respect to local perturbations of the input. This metric is important because it is becoming a consensus that more stable explanations tend to be more reliable [98]. The original metric considers a neighborhood size ϵ of a point t to quantify stability based on the input features. Here we adapt this metric for regression by using the same definition of neighborhood used for NF - as we work with prototypes, the neighborhood has size one - and define global stability (GS) as the median of the local stability (LS) of each example, as shown in Eq. 5.4. Note that while fidelity considers the differences in the outputs, stability looks at the values of the input features.

$$GS(t, PR_t) = median_{\forall t \in T}(LS(t, PR_t)) \tag{5.4}$$

In this case, distance(x_i, x_j) is defined as the Ls norm (also called Minkowski distance) with $s = 0.3$, as shown in Equation 5.5.

$$LS(x_i, x_j) = dist_{Mink}(x_i, x_j) = (\sum_{a=1}^{d} |x_{ia} - x_{j_a}|^{0.3})^{1/0.3} \tag{5.5}$$

where x_i are the features of the instance to be explained and x_j are the features of the explaining prototype.

The value 0.3 was set after preliminary evaluation, considering that in [5] the authors evaluated the behavior of the Ls norm and observed that smaller values are more appropriate to high dimensional data.

5.1.2 Strategies for Assigning Prototypes to Test Points

As shown in Algorithm 1, we first select the set of prototypes (line 2) – which are able to provide global explanations – and later use them to provide local explanations (lines 6-7). When providing explanations, we associate the prototype closer to the instance to be explained using a distance measure. As previously pointed out, for regression problems it is essential to use both the input variables as well as the predicted value.

Both Local Fidelity (LF) and Local Stability (LS) measure different aspects of the prototypes found, and hence we propose to combine them to assign a prototype to a test

example. Note that the global model will be found using both the error and these metrics. As already explained, it is not clear if one of these metrics should be more important than the other. Hence, after we found the set of representative prototypes, we use the L2 of the normalized values of LF and LS to assign test instances of prototypes, as shown in Eq. 5.6.

$$L2FS(f, PR_{t'}, t') = \sqrt{\left[\frac{1}{\log(\overline{LF}(f(PR_{t'}), f(t')))}\right]^2 + \left[\frac{1}{\log(\overline{LS}(t', PR_{t'}))}\right]^2} \tag{5.6}$$

$\overline{LF}$ and $\overline{LS}$ are normalized values by the min-max of the original metrics. In addition, in order to avoid outliers, the logarithmic function was applied.

5.2 Experimental Setup

As previously mentioned, M-PEER is model-agnostic. For the purpose of this work, we chose the Random Forest Regressor (RF) method to perform experiments, as it is among the state-of-the-art methods for regression. We used the implementation from the Python Scikit Learn library[1]. Note that although the individual models provided by the Random Forest algorithm can be considered inherently interpretable, the level of interpretability is considerably reduced as the number of trees used by the model increases. We consider that with approximately 10 trees the model provided is already similar to any black-box model, and difficult for a human to understand. Therefore, the number of trees used was defined as 10 and all other parameters were kept as default. M-PEER was implemented using DEAP's SPEA2 implementation [37].

After preliminary parameter tuning, M-PEER was run with a population of 300 individuals evolving for 75 generations. The probabilities of crossover and mutation were defined as 0.6 and 0.4, respectively, and the tournament size was set to 2. Another important parameter of M-PEER is the number of prototypes returned. Considering the limited storage capacity in human working memory while processing information, and previous research showing a limit quantification of approximately 7 elements [84], we analyze the selection of 5 and 10 prototypes per regression model built.

We evaluated the proposed strategies on 25 real-world datasets [28, 20, 8, 23], described in Table 5.1. All variables, including the output y, were standardized to have mean equal to zero and variance equal to one.

[1] Available in https://scikit-learn.org/

Table 5.1: Dataset used in the experiments of the DELA explanation method (Chapter 5) and the HE explanation method (Chapter 6).

Id	Dataset	# Attributes	# Train	# Test
1	cholesterol	14	209	88
2	auto-mpg	8	279	119
3	sensory	12	403	173
4	strike	7	437	188
5	day_filter	12	510	221
6	qsar_fish	7	636	272
7	concrete	9	721	309
8	music	70	741	318
9	house	80	1022	438
10	wineRed	12	1279	320
11	communities	103	1395	598
12	crimes	103	1550	664
13	abalone	8	2924	1253
14	wineWhite	12	3918	980
15	cpu_act	22	5734	2458
16	bank32nh	33	5734	2458
17	puma32H	33	5734	2458
18	compactiv	21	5734	2458
19	tic	85	6876	2947
20	ailerons	41	9625	4125
21	elevators	19	11619	4980
22	california	8	14448	6192
23	house_16H	17	15949	6835
24	fried	11	28538	12230
25	mv	10	28538	12230

5.2.1 Baselines

We compared the proposed approach with 4 baselines: two adaptations of machine learning algorithms that, to the best of our knowledge, were not used before for explainability: a cluster-based and a hub-based algorithm; a single-objective evolutionary algorithm - with two different versions of a fitness function; and ProtoDash, a method considered state-of-the-art in terms of providing explanability of agnostic models in regression.

We use the k-medoids as a representative of clustering algorithms. Given two instances $p_i = (\mathbf{x}_i, y_i)$ and $p_j = (\mathbf{x}_j, y_j)$, we used the Minkowski distance or s norm with $s = 0.3$, according to Eq. 5.5. We have also tested other metrics, including the Euclidean distance, and the best results were obtained with the aforementioned metric.

The k-medoids implementation was the one from Scikit-Learn-Extra[2], also with its default parameters but the distance function.

We also compared our approach with a method based on the concept of Hub points [74]. Hub points are training instances that appear in the lists of n nearest neighbors of other instances much more frequently than others. For calculating the Hub points, we need to define the number of closest neighbors considered. In preliminary tests we found that, by varying the value of the number of closest neighbors between 1 and 13, in intervals of 2, the best results were found using 11 neighbors. Hence, this is the neighborhood size used in our baselines. We kept the Minkowski distance to calculate neighbors, as other metrics were tested and presented no statistically significant difference from the aforementioned.

As we are proposing a multi-objective GA, it makes sense to evaluate the impact of the multi-objective optimization. Hence, we compare the results with those of a single-objective GA. We used a canonical version of a GA, using single-point crossover, single point mutation and tournament selection. Different values of fitness were tested: a linear combination of global stability and global fidelity (see Eq.5.7), from now on referred as GA (GFS), and the RMSE - generating GA (RMSE). When using the RMSE, the algorithms resembles methods for instance selection, where examples are chosen based solely on error. After preliminary parameter tuning, the GA was executed with a population of 600 individuals evolved for 150 generations. The probabilities of crossover and mutation were defined as 0.8 and 0.2, respectively, and the tournament size was set to 2. Note that although the GA performs more evaluations that M-PEER, the results show that even in this disadvantageous scenario, M-PEER performs better than the single-objective methods.

For ProtoDash we use the implementation available in [13] to find the prototype points. We used the same parameters used by Protodash in the original paper: the Gaussian kernel with sigma equal to 2.

5.2.2 Evaluation

Considering the non-deterministic nature of the results obtained by RF, GA (GFS), GA (RMSE), and M-PEER (multi-objective), all results reported are medians over 30 executions. All methods proposed and evaluated in this work are freely available for download[3].

[2]Available in https://scikit-learn-extra.readthedocs.io/en/latest/

[3]Available in: https://github.com/renatomir/MPEER-TELO2022

To evaluate the quality of the individual, composed of the e prototypes selected by the evolutionary method, we propose to combine the global version of the adapted metrics, i.e., GF and GS, as shown in Eq. 5.7.

$$GFS(f, PR'_t, t') = \sqrt{\left[\frac{1}{\log(\overline{GF}(f, PR'_t, t'))}\right]^2 + \left[\frac{1}{\log(\overline{GS}(t', PR'_t))}\right]^2} \tag{5.7}$$

Note that, in the fitness minimization process, the GF, GS, and $RMSE$ objectives are calculated taking into account the explanation of the training points t. However, in this evaluation step the calculation will be carried out taking into account the test points t'. In addition, $\overline{GS}$ and $\overline{GF}$ are normalized values of the original metrics using min-max. Again, in order to avoid outliers that could skew the results considering only global stability or global fidelity, the logarithmic function is applied.

Apart from running M-PEER with the three objectives previously defined, we have also tested it with two other combinations of objectives: i) Global Stability and Global Fidelity, as defined in Eq. 5.4 and 5.3 respectively; ii) GFS and RMSE (Eq. 5.7 and 5.1).

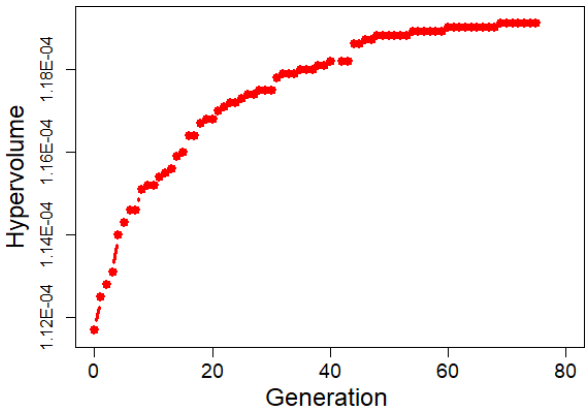
5.3 Experimental Results and Discussion

This section presents the results of M-PEER. We first look at the behavior of the method in terms of search and convergence with different sets of objectives. Next, we compare M-PEER with the other methods previously proposed to solve this task and discuss the results in terms of error, stability and fidelity.

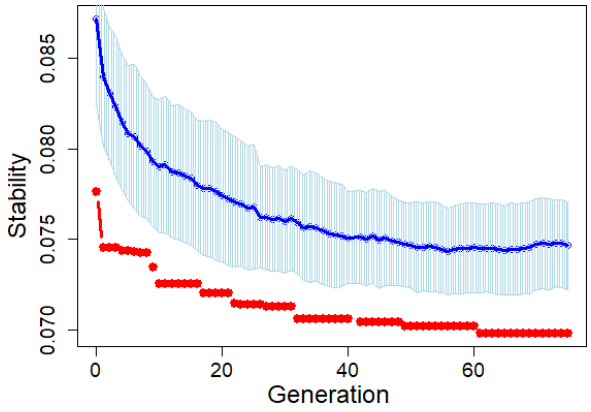
5.3.1 Search and Convergence

As previously mentioned, we implemented three versions of M-PEER considering two or three objectives, namely: i) global Stability and global Fidelity (S+F); ii) GFS and RMSE (GFS+E); and iii) global Stability, global Fidelity and RMSE (S+F+E). Ultimately, we want to optimize the three objectives, but there might be a correlation between these metrics. As searching with two objectives is usually simpler than searching with three, the three combinations above were considered.

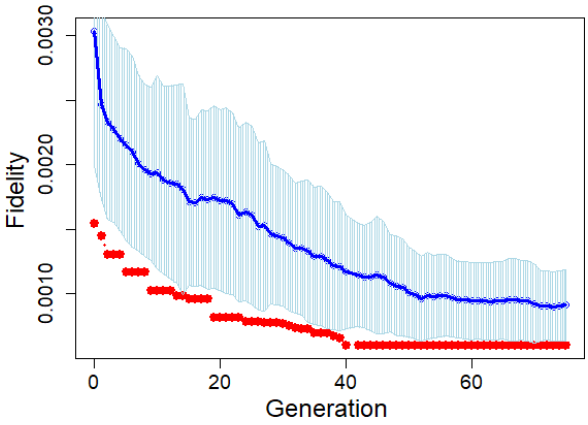
Figure 5.2: Convergence of M-PEER in the scenario of minimizing the objectives of Global Stability, Global Fidelity and RMSE (choice of 10 prototypes - first run - MV dataset). In Figures b, c and d, the line in red represents the minimum value found in the population, the dark blue line represents the average value and the shading blue represents the standard deviation.



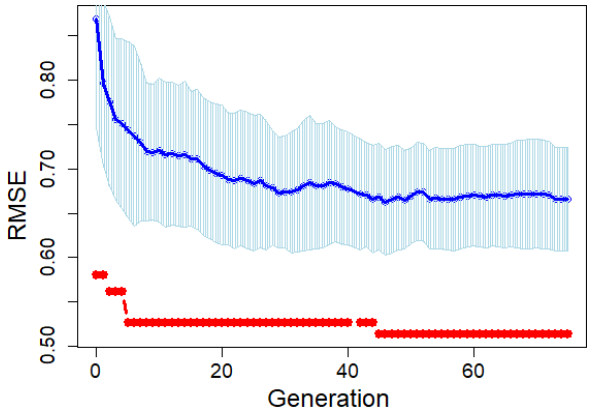
(a) Hypervolume



(b) Global Stability



(c) Global Fidelity



(d) RMSE

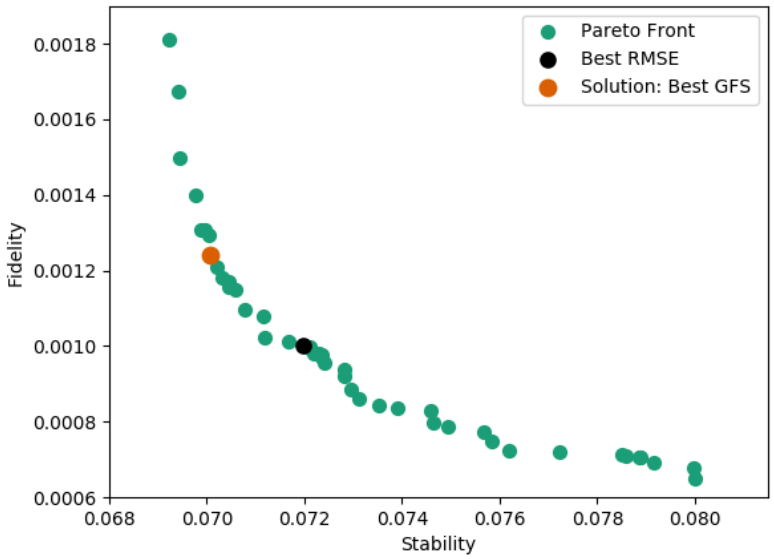
Source: created by the author.

In the case of multi-objective algorithms, the effectiveness of the search can be measured using the hypervolume of the Pareto front generated [125, 123]. The classic definition of hypervolume is based on the hypercube formed by the Pareto dominance over a fixed point and is defined as follows. Given a set of points S $\subset \mathbb{R}^d$ and a reference point $\alpha \in \mathbb{R}^d$, the hypervolume indicator of S is the measure of the region weakly dominated by S and bounded above by α [47], as defined in Eq. 5.8. The higher its value, the better the quality of the solution set evolved.

$$H(S) = \forall(q \in \mathbb{R}^d | \exists p \in S : p \leq q \, and \, q \leq \alpha) \tag{5.8}$$

Since all versions with different objectives had similar behaviors, we show only the behavior for S+F+E for dataset MV considering 10 prototypes as explanations. Figure 5.2a shows the hypervolume formed by the Pareto front considering as reference point

Figure 5.3: Final population of M-PEER in the scenario of minimizing the objectives of Global Stability and Global Fidelity (choice of 10 prototypes - first run - MV dataset)

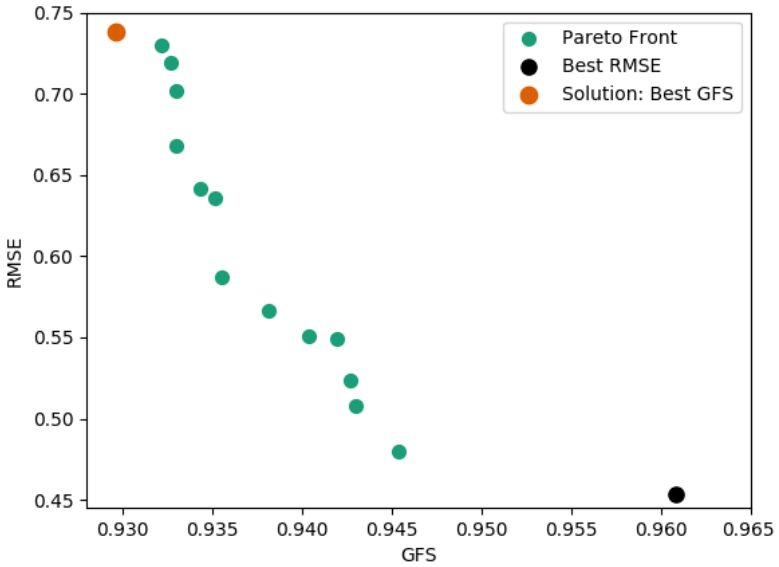


Source: created by the author.

[0.103, 0.008, 1.283], where the values represent, respectively, the maximum stability, maximum fidelity and maximum RMSE among the individuals in the initial population in this run. Although this is an ad-hoc choice, it makes sense if we consider that the value of these metrics in the Pareto front will not become worse than their values in the first generation. As expected, the Pareto front with the best solutions approaches the origin [0,0,0] over the generations, making the value of the hypervolume grow consistently with search progression. Figures 5.2b, 5.2c and 5.2d show that the values of the three objectives being optimized are improving over time for the best solution found, identified using Eq. 5.7. In these figures, the red line shows the minimum value of the evaluated objective in the population, the dark blue line the mean value and the blue shading the standard deviation, showing the dispersion of values in the current population.

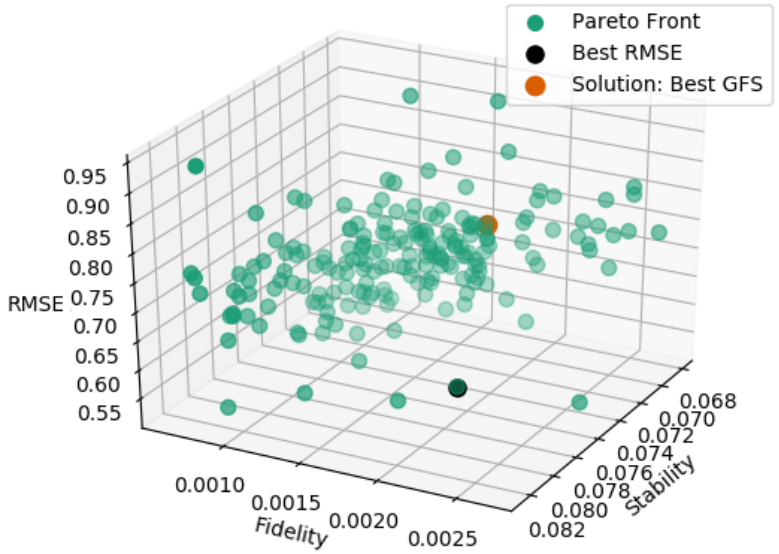
Figures 5.3, 5.4 and 5.5 show the dispersion of the prototype sets belonging to the population at the end of a specific execution of the algorithm. As we can see, in all scenarios presented, regardless of the number of objectives of the problem, the best solutions converge to the origin of the Cartesian plan. We also observe that the metric proposed to select the best individual from the Pareto front (GFS) produces a good balance between the importance of fidelity and stability while accounting for the error, as opposed to choosing the individual with the smallest error or the best values for one of these metrics. The result in Figure 5.5 gives us a strong indication of the correlation between the fidelity and stability objectives and the RMSE as, although the error is not verified during the assignment process, we see that the selected prototype set had an intermediate value of error among the solutions found.

Figure 5.4: Final population of M-PEER in the scenario of minimizing the objectives of GFS and RMSE (choice of 10 prototypes - first run - MV dataset)



Source: created by the author.

Figure 5.5: Final population of M-PEER in the scenario of minimizing Stability, Fidelity and RMSE (10 prototypes - first run - MV dataset).



Source: created by the author.

5.3.2 Comparison between strategies for selecting prototypes

This section presents the results of RMSE (Eq. 5.1) and GFS (Eq. 5.7) obtained by the proposed strategies and the comparison methods to select prototypes. We show the results considering 5 and 10 prototypes in Tables 5.2 and 5.3 for RMSE and in Tables 5.4 and 5.5 for GFS. Apart from the methods described in Section 5.2.1, we also added to our experiments a naive baseline, which chooses prototypes randomly from the training

Table 5.2: Results for RMSE when providing explanations with 5 prototypes.

	RMSE								
Dataset	**Random**	**Kmedoids**	**Hub-11**	**GA (GFS)**	**ProtoDash**	**M-PEER (S+F)**	**M-PEER (GFS+E)**	**GA (RMSE)**	**M-PEER (S+F+E)**
1	0.6480	0.7704	0.7598	0.6401	0.9029	0.6311	0.6299	**0.5301**	0.6285
2	0.5747	0.6834	0.7836	0.6245	0.6939	0.6207	0.6203	**0.4753**	0.6194
3	0.7048	0.8496	1.1039	0.6872	1.1082	0.7004	0.6545	**0.6323**	0.6754
4	0.6160	0.6087	0.9055	0.6335	0.7243	0.6315	0.5757	**0.4978**	0.5808
5	0.75426	0.8286	0.8080	0.6602	0.8128	0.6714	0.6626	**0.6511**	0.6671
6	0.65982	0.7910	0.8385	0.6630	0.6787	0.6528	0.5959	**0.5725**	0.6037
7	0.85239	0.8676	1.3906	0.8317	1.4912	0.8148	0.8054	**0.7272**	0.8180
8	0.81525	0.8380	0.8179	0.7910	0.7931	0.7940	0.7958	**0.7793**	0.7899
9	0.80990	0.8124	1.0864	0.7849	0.9466	0.7531	0.7148	**0.6926**	0.7054
10	0.63630	0.8847	1.0243	0.5776	0.6100	0.5792	**0.5712**	0.5859	0.5738
11	0.68084	0.7434	0.6753	0.6561	1.0618	0.6423	0.6368	**0.6336**	0.6431
12	0.40764	0.4075	0.4097	0.4072	0.4087	0.4085	**0.4029**	0.7768	0.4065
13	0.66831	0.7946	0.7493	0.6870	0.8224	0.7078	0.6305	**0.5834**	0.6359
14	0.69425	0.7715	0.6989	0.6723	1.4315	0.6668	0.6524	**0.6134**	0.6640
15	0.97125	0.9482	1.0511	0.9701	0.9364	0.9858	0.9584	**0.9123**	0.9562
16	0.91740	0.9097	0.7861	0.9629	1.0166	0.9679	0.9504	**0.7682**	0.9539
17	1.00295	1.0592	1.0299	0.9914	1.2153	0.9871	0.9930	1.0039	**0.9834**
18	0.94263	0.8906	1.1068	0.9494	1.0005	0.9315	0.9168	**0.8641**	0.9178
19	**0.73499**	0.7399	0.9862	0.7350	0.9438	0.7350	0.7350	0.8352	**0.7350**
20	0.78799	0.7337	1.0664	0.8984	0.9725	0.8463	0.8893	**0.6837**	0.8777
21	0.90729	0.8502	0.9807	0.9237	0.7599	0.9165	0.9059	**0.7398**	0.9165
22	0.90526	0.9232	1.0087	0.8992	0.9015	0.9398	0.8801	**0.7609**	0.9196
23	0.84867	0.8522	0.9777	0.8433	1.0405	0.8356	0.8215	**0.7370**	0.8264
24	0.91740	0.9647	0.9878	0.8971	0.9612	0.8951	0.8666	**0.8251**	0.8933
25	0.92184	0.9747	0.9488	0.9106	1.2479	0.9367	0.9008	**0.8331**	0.9194

set. To be fair with the multi-objective methods, the random method selects, for each scenario, the best subset of prototypes among 300×75 randomly selected instances.

Note that, for the results of RMSE, the best values is usually obtained by GA (RMSE). This is expected, as GA only accounts for RMSE in its objective function. For GFS, one of the versions of M-PEER usually obtains the best results. However, there is one exception. For the very small datasets (the ids of the datasets are ordered according to the number of training instances), shown in the first lines of the tables, the random approach outperforms all other methods. This happens since the number of time instances are randomly selected 5 by 5 or 10 by 10 (22,500). In this case, randomly selecting instances with high GFS is "easy". However, note that the error in this case is still superior for the random method. That is exactly why we are proposing a method that can optimize both objectives simultaneously. Hence, for datasets with few training instances, the excessive effort of an evolutionary method for choosing prototypes is not worth spending. Thus, in the next results shown in this section we will focus only on datasets with at least 500 instances in the training set, considering the limitations of the proposed method in this scenario.

In order to make these results more comparable in datasets with more than 500 instances in the training set, we show the critical diagrams [24] for the RMSE evaluation metrics in Figures 5.6 and for GFS evaluation metrics in Figures 5.7. These diagrams were generated after carrying out an adapted Friedman test followed by a Nemenyi post-hoc

Table 5.3: Results for RMSE when providing explanations with 10 prototypes.

	RMSE								
Dataset	**Random**	**Kmedoids**	**Hub-11**	**GA (GFS)**	**ProtoDash**	**M-PEER (S+F)**	**M-PEER (GFS+E)**	**GA (RMSE)**	**M-PEER (S+F+E)**
1	0.6727	0.7820	0.7066	0.6378	0.7351	0.6646	0.5735	**0.5008**	0.6094
2	0.4667	0.5654	0.7324	0.5079	0.4344	0.5199	0.4294	**0.3515**	0.4557
3	0.7128	0.8242	0.9037	0.6806	0.8568	0.7203	0.6558	**0.6229**	0.6629
4	0.5438	0.6329	0.9074	0.5634	0.6347	0.5463	0.5204	**0.5043**	0.5178
5	0.6459	0.6919	0.7495	0.6450	0.5811	0.6253	0.6002	**0.4735**	0.6015
6	0.5824	0.6228	0.7137	0.5410	0.5880	0.5480	0.5032	**0.4806**	0.5147
7	0.7250	0.7951	1.2511	0.7115	1.6273	0.7091	0.6496	**0.6113**	0.6779
8	0.8256	0.8119	**0.7717**	0.8160	0.7862	0.8110	0.7879	0.7771	0.8024
9	0.6848	0.7288	1.0989	0.6200	0.8035	0.6419	0.6075	**0.5553**	0.6258
10	0.6280	0.6505	1.0386	0.5602	0.6303	0.5840	0.5261	**0.5083**	0.5440
11	0.6491	0.6914	0.8270	0.6441	1.0240	0.6552	0.6381	**0.5747**	0.6339
12	0.4017	**0.3920**	0.4075	0.4061	0.4022	0.4041	0.3990	1.2990	0.4044
13	0.6338	0.6961	0.7276	0.7018	0.8214	0.6865	0.6068	**0.5407**	0.6216
14	0.6656	0.7455	0.7347	0.6445	1.0828	0.6084	0.6282	**0.5497**	0.5991
15	0.9338	0.8829	0.9645	0.8951	0.9287	0.9150	0.7498	**0.4178**	0.7513
16	0.8619	0.8173	0.8272	0.9563	0.9327	0.9674	0.9323	**0.6526**	0.9560
17	1.0151	1.0830	1.0635	0.9804	0.9780	0.9836	0.9684	**0.9236**	0.9752
18	0.8814	0.8700	1.0859	0.8965	0.8806	0.8584	0.6894	**0.4049**	0.7347
19	0.7350	0.8439	1.0308	0.7350	0.9617	0.7350	0.7350	**0.7249**	0.7350
20	0.7675	0.7030	0.8111	0.7496	0.9800	0.7271	0.7054	**0.5787**	0.7068
21	0.8376	0.8171	0.9660	0.9084	0.8299	0.8788	0.8530	**0.6555**	0.8696
22	0.7849	0.8317	1.1208	0.7857	0.8240	0.8072	0.7601	**0.6268**	0.7846
23	0.7699	0.7818	0.9182	0.8146	0.9398	0.7930	0.7893	**0.6581**	0.7921
24	0.8481	0.8921	0.8390	0.7985	0.8990	0.7717	0.7603	**0.7045**	0.7499
25	0.8290	0.8238	0.9163	0.7748	1.0434	0.7684	0.7418	**0.6930**	0.7450

Table 5.4: Results for GFS when providing explanations with 5 prototypes.

	GFS								
Dataset	**Random**	**Kmedoids**	**Hub-11**	**GA (GFS)**	**ProtoDash**	**M-PEER (S+F)**	**M-PEER (GFS+E)**	**GA (RMSE)**	**M-PEER (S+F+E)**
1	**0.7391**	0.8148	0.7935	0.7761	0.8308	0.7681	0.7712	0.8514	0.7744
2	**0.7768**	0.8371	0.9135	0.7888	0.8920	0.7901	0.7860	0.8772	0.7862
3	**0.8499**	0.9538	1.0378	0.8875	0.9936	0.8870	0.8847	0.9412	0.8912
4	**1.0863**	1.2361	1.3269	1.0974	1.3753	1.0913	1.0885	1.3083	1.0898
5	0.8954	0.9632	0.9650	**0.8845**	0.9520	0.8890	0.8897	1.0479	0.8878
6	0.8136	0.8696	0.9906	0.8185	0.9424	0.8151	0.8186	0.9196	**0.8124**
7	1.0620	1.1171	1.5172	**1.0454**	1.6713	1.0590	1.0468	1.2894	1.0583
8	1.1332	1.1541	1.1598	1.1307	1.1693	**1.1281**	1.1288	1.2370	1.1301
9	0.7447	0.7499	0.8682	0.7283	0.8796	0.7271	0.7285	0.8494	**0.7255**
10	1.1026	1.1604	1.1210	1.0926	1.2280	**1.0905**	1.0972	1.1992	1.0918
11	1.2560	1.2974	1.2792	1.2334	1.3757	1.2332	1.2379	1.4120	**1.2321**
12	1.0610	1.0666	1.0630	**1.0393**	1.1081	1.0471	**1.0393**	1.2383	1.0417
13	0.7847	0.8048	0.8419	0.7734	0.9618	0.7768	0.7748	0.8915	**0.7731**
14	1.1231	1.1387	1.1356	**1.1075**	1.2572	1.1103	1.1097	1.2182	1.1077
15	0.7025	0.7161	0.7202	0.6884	0.7879	**0.6861**	0.6883	0.8186	0.6882
16	1.7019	1.7263	1.6549	**1.6296**	1.6700	1.6320	1.6298	1.8884	1.6310
17	1.8469	2.0073	1.7972	**1.7648**	1.9603	1.7667	1.7670	1.9804	1.7736
18	0.7153	0.7331	0.7894	0.7012	0.7889	0.7002	0.7016	0.8160	**0.6997**
19	0.7033	0.7163	0.8287	0.6789	0.7565	**0.6786**	0.6789	0.8078	0.6813
20	0.7019	0.7128	0.9283	0.6789	0.7671	**0.6750**	0.6763	0.7992	0.6773
21	0.7212	0.7422	0.7865	0.7157	0.7775	**0.7144**	0.7150	0.8428	0.7144
22	0.8455	0.8737	0.9072	0.8272	0.9654	0.8278	0.8279	0.9301	**0.8260**
23	0.8028	0.8005	0.8371	0.7821	0.8840	**0.7808**	0.7826	0.8742	0.7811
24	1.5953	1.6426	1.6214	1.5648	1.6821	**1.5639**	1.5667	1.7318	1.5655
25	1.0215	1.0597	1.0614	0.9918	1.1780	**0.9886**	0.9932	1.1189	0.9913

with a significance level of 0.05. As we are comparing multiple approaches, Bonferroni's correction was applied to all tests [24]. In these diagrams, the main line shows the

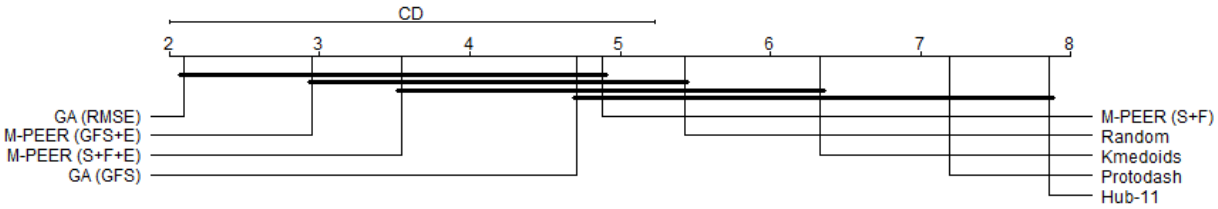
Table 5.5: Results for GFS when providing explanations with 10 prototypes.

	GFS								
Dataset	**Random**	**Kmedoids**	**Hub-11**	**GA (GFS)**	**ProtoDash**	**M-PEER (S+F)**	**M-PEER (GFS+E)**	**GA (RMSE)**	**M-PEER (S+F+E)**
1	**0.6998**	0.7811	0.7293	0.7235	0.7785	0.7238	0.7230	0.7860	0.7231
2	**0.7153**	0.7701	0.8044	0.7335	0.7782	0.7341	0.7339	0.7679	0.7332
3	**0.7528**	0.8579	0.8761	0.8175	0.8457	0.8150	0.8153	0.8389	0.8109
4	**0.8898**	1.0525	1.0260	0.9090	1.1911	0.9071	0.9148	1.0518	0.9152
5	**0.8044**	0.8731	0.8809	0.8092	0.8386	0.8079	0.8072	0.9058	0.8134
6	0.7594	0.8064	0.9199	0.7565	0.8605	0.7545	0.7559	0.8312	**0.7528**
7	0.9482	0.9943	1.1456	0.9336	1.5760	0.9372	**0.9330**	1.1289	0.9399
8	1.0956	1.1202	1.1191	1.0886	1.1247	**1.0844**	1.0892	1.1776	1.0868
9	0.7172	0.7284	0.8159	0.6991	0.8169	0.6992	**0.6989**	0.7884	0.7005
10	1.0537	1.1039	1.0711	1.0355	1.1534	1.0340	**1.0336**	1.1228	1.0356
11	1.1996	1.2115	1.2218	1.1710	1.2846	1.1689	1.1714	1.3110	**1.1674**
12	1.0175	1.0217	1.0101	**0.9954**	1.0712	0.9985	0.9968	1.1459	1.0010
13	0.7286	0.7396	0.7651	0.7168	0.8491	**0.7112**	0.7114	0.7867	0.7125
14	1.0700	1.1106	1.0783	1.0563	1.1255	1.0509	1.0517	1.1366	**1.0502**
15	0.6806	0.6937	0.7008	0.6626	0.7558	**0.6606**	0.6620	0.7686	0.6617
16	1.6153	1.7067	1.5541	1.5340	1.5690	1.5355	1.5341	1.7417	**1.5323**
17	1.7372	1.8488	1.6810	1.6556	1.8157	**1.6531**	1.6554	1.8306	1.6572
18	0.6924	0.7071	0.7736	0.6735	0.7281	**0.6708**	0.6724	0.7762	0.6711
19	0.6728	0.6902	0.7204	0.6431	0.7072	**0.6427**	0.6435	0.7489	0.6447
20	0.6324	0.6536	0.7016	0.5983	0.6964	0.5961	**0.5946**	0.7336	0.5985
21	0.6643	0.6855	0.6874	0.6428	0.7421	**0.6386**	0.6395	0.7749	0.6407
22	0.8010	0.8173	0.8696	0.7802	0.8821	**0.7787**	**0.7787**	0.8562	0.7798
23	0.7807	0.7764	0.7937	0.7550	0.8431	**0.7524**	0.7535	0.8213	0.7538
24	1.4953	1.5451	1.4945	1.4667	1.6064	1.4645	1.4656	1.5728	**1.4644**
25	0.9551	0.9822	0.9673	0.9302	1.0931	**0.9252**	0.9292	1.0072	0.9274

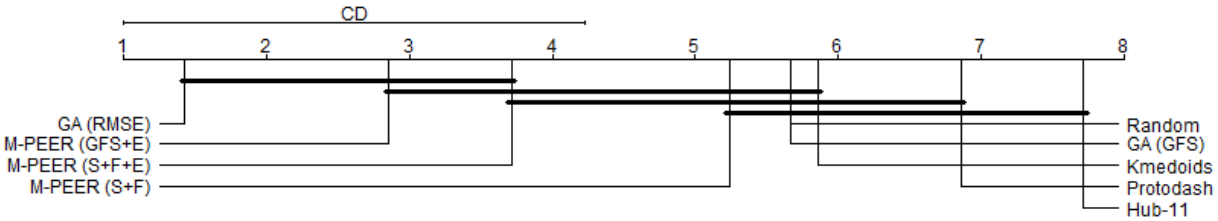
average ranking of the methods, i.e., how well one method did when compared to the others. This ranking takes into account the absolute value obtained by each method according to the evaluated metric. The best methods are shown on the left (lowest rankings). The critical difference interval (CD in the plots) is determined by the Friedman test according to the significance level defined. If the average difference between two algorithms is greater than CD, then the null hypothesis that the algorithms have the same performance is rejected. Finally, the diagram connects groups of methods that do not present statistically significantly differences. Note that the size of the lines connecting the methods corresponds to the size of the CD interval.

Error is the basic quality of explanation metric by which most current methods are evaluated. Recall that the RMSE accounts for the distance of one singles regression model trained with two different sets of instances: the complete training set and the selected prototypes. As we can see in the critical diagram of Figure 5.6a, when selecting 5 prototypes, all evolutionary methods managed to obtain the best results, and it is not possible to say there is statistically significant differences between them. For 10 prototypes (Figure 5.6b), the statistically significant superior methods were those based on evolutionary approaches whose fitness takes into account the error, i.e., GA RMSE (the minimization objective was only the RMSE), M-PEER GFS+E, and M-PEER GFS+F+E. Regardless of selecting 5 or 10 prototypes, the methods with lower performance in relation to the RMSE metric were HUB-11, ProtoDash and Kmedoids.

Figure 5.6: Critical diagrams of methods considering RMSE. From left to right, methods are ranked according to their performance, and those connected by a bold line present no statistical difference in their results.



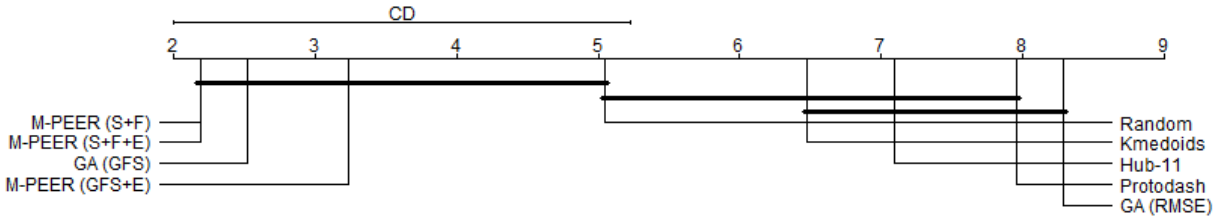
(a) Selection of 5 prototypes.



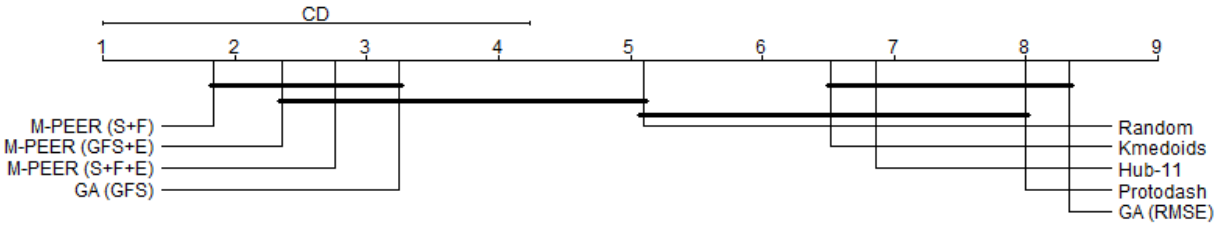
(b) Selection of 10 prototypes.

Source: created by the author.

Figure 5.7: Critical diagrams of methods considering GFS. From left to right, methods are ranked according to their performance, and those connected by a bold line present no statistical difference in their results.



(a) Selection of 5 prototypes.



(b) Selection of 10 prototypes.

Source: created by the author.

However, it is not enough to have only a small error. The selected prototypes must also have a "semantic meaning" in the context of the features and the output, which can explain the model as a whole. So, in addition to the error, we also want the GFS metric to be minimized. We can observe from the critical diagram in Figure 5.7b that, in the selection process of 10 prototypes, the following methods were statistically superior: M-

PEER S+F, M-PEER GFS+E, M-PEER S+F+E, and the GA minimizing only the GFS metric. As for the selection of 5 prototypes, the random choice also managed to obtain good results, staying at the limit of the intervals of the statistical difference. In both scenarios, the worst methods considering the GFS metric were GA RMSE, ProtoDash and HUB-11.

Thus, we can conclude that the methods whose results stood out for both metrics were M-PEER GFS+E and M-PEER S+F+E. We also advise the use of evolutionary methods for datasets with at least 500 examples. Although this number is arbitrary, our experiments showed this as a good trade-off between the time for running the method and the results obtained.

5.3.3 ProtoDash vs. M-PEER

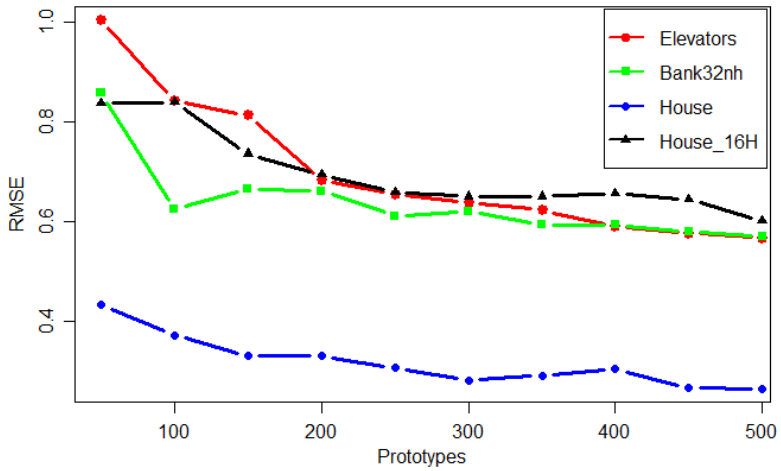
This section discusses and compares the results of ProtoDash and M-PEER considering two different dimensions: the error (RMSE) and the explanations provided. We start by looking at the RMSE.

As mentioned earlier, the work where ProtoDash is presented compares different methods of selecting prototypes for the regression problem with a fixed number of prototypes. This number corresponds to the knee point in the RMSE versus number of prototypes curve (see Figure 5.8), where RMSE improvements become stable. The authors of Protodash do not discuss how the knee point was selected. Here, we choose this value considering RMSE only. We first vary the number of prototypes from 50 to 500, in steps of 50. We then select the knee point according to the RMSE improvement in subsequent iterations. The knee is selected as the point where: (i) there is a lower RMSE improvement from iterations t to t+1 than the improvement obtained from iterations t-1 to t; (ii) the cumulative RMSE improvement from iteration 1 (50 prototypes) to t is greater than the possible cumulative RMSE improvement from iteration t+1 until the last iteration (500 prototypes). Note that this second criteria verifies if at least half of the possible RMSE improvement has already happened at the candidate knee point.

Considering the way the number of prototypes is chosen by ProtoDash, we selected four of our datasets to carry out a similar study and compared the results obtained by our two best M-PEER versions with ProtoDash. Figure 5.8 shows the curves as we increase the number of prototypes chosen. The knee points chosen for these four datasets were: i) 200 for the Elevators dataset; ii) 100 for the Bank32nh dataset; iii) 200 for the House_16H dataset; and iv) 150 for the House dataset.

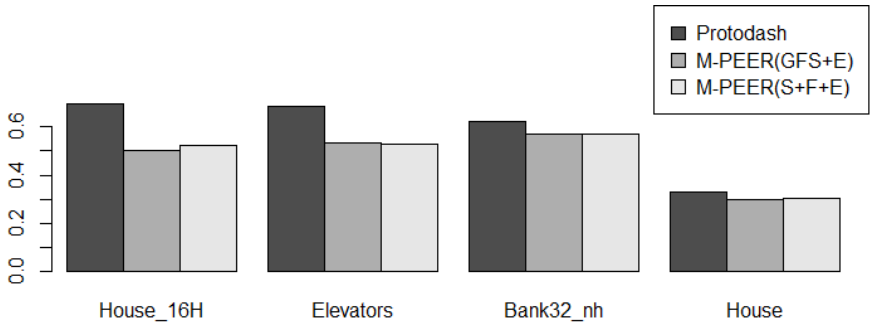
We believe these values are far too large for a human to be able to analyze the

Figure 5.8: Number of prototypes chosen, represented by the point where the RMSE objective improvement becomes negligible.



Source: created by the author.

Figure 5.9: RMSE in the point where improvement in the RMSE objective becomes incremental.



Source: created by the author.

prototypes obtained and extract useful information from the model built. As discussed before, the human working memory is extremely limited and a value much greater than 7 could make the explanation unproductive.

The results obtained are shown in the graph of Figure 5.9. Note that the M-PEER version with GFS and RMSE objectives (GFS+E), in general, obtained the best results. Furthermore, Protodash was worse in all evaluated scenarios. When comparing these results to the ones obtained using a much smaller number of prototypes, as shown in Tables 5.2 and 5.3, we observe they are consistent for explanation scenarios, i.e., ProtoDash obtains, in general, inferior results than the evolutionary approaches both in terms of the error and the proposed GFS metric, for any number of selected prototypes.

But what about the explanations generated? Do these methods agree on them? Comparing explanations given by different methods is one of the hardest things when proposing new methods for explainability. We have adapted two metrics proposed in the classification context to do so, but evaluating how well they perform when compared to what a human calls interpretable is hard.

Table 5.6: Global explanation (size of prototype set = 5) selected by different explainability methods for the auto-mpg dataset.

	Input features							Output MPG	
	Cylinders	Displacement	Horsepower	Weight	Acceleration	Model year	Origin	y	y'
	4	97	88	2130	14.5	71	3	27	27.2
	8	304	150	3672	11.5	73	1	14	15
ProtoDash	4	91	53	1795	17.4	76	3	33	32.4
	4	90	70	1937	14.2	76	2	29	29.1
	4	85	65	2110	19.2	80	3	40.8	38.89
Prototypes variance	3.2	9113.3	1470.7	582507.7	8.973	11.7	0.8	95.99	76.86
	6	225	95	3264	16	75	1	19	19.37
M-PEER(GFS+E)	4	98	90	2265	15.5	73	2	26	26.25
and	8	318	150	4096	13	71	1	14	13.60
M-PEER(S+F+E)	4	140	88	2720	15.4	78	1	25.1	25.06
	4	91	67	1965	15.7	82	3	32	33.50
Prototypes variance	3.2	9289.3	959.5	716385.5	1.457	18.7	0.8	47.79	56.26

In order to give an insight on how the explanations are given to users, Table 5.6 shows the set of prototypes selected to explain the test instances by ProtoDash, M-PEER(GFS+E), and M-PEER(S+F+E) for the dataset *auto-mpg*, which describes the characteristics of cars and relates them to the result of gas consumption in miles per gallon (MPG). We chose this dataset for two reasons: the input attributes are easy to interpret and consumption varies depending on the car's construction engineering, something well-known in the specialized literature but ordinary enough that it can be interpreted with a low degree of expertise in the problem.

As we can see from Table 5.6, the M-PEER versions selected exactly the same prototypes to explain the model when considering 5 prototypes. We can also observe that the features with the greatest variance among the selected datasets (reported in the row labeled "prototypes variance") were *Weight* and *Displacement*. According to [33], there is a well-known relationship between the engine displacement and weight features and the consumption of a car. The weight has a linear relationship with consumption. Displacement is inversely proportional: when we decrease the engine displacement of the car, we are increasing its consumption. Given these known relationships, we consider the great variability found in the values of the prototypes for the features weight and displacement is appropriate, as they make the set of prototypes able to better cover the space of solutions of the different values for consumption and thus better represent the model.

Furthermore, we can also analyze the diversity of selected prototypes. If we calculate the pairwise distance between the 5 prototypes selected by the algorithms (Euclidean distance or Minkowski distance with parameter 0.3) we can see that larger values are obtained by the M-PEER versions, showing they obtain a greater coverage of the problem solution space and, consequently, greater diversity in the selected prototypes.

Finally, Table 5.7 presents the explanatory prototype assigned to a specific test point. Again, if we calculate the distances between the evaluated point and the explanations (Euclidean distance or Minkowski distance with parameter 0.3), we can observe

Table 5.7: Local explanation provided by the methods for a specific test instance.

	Input features							Output MPG	
	Cylinders	Displacement	Horsepower	Weight	Acceleration	Model year	Origin	y	y'
Test instance	4	98	70	2125	17.3	82	1	36	34.1
ProtoDash	4	91	53	1795	17.4	76	3	33	32.4
M-PEER(S+F+E)	4	91	67	1965	15.7	82	3	32	33.5

Table 5.8: Execution time for the methods in the smallest and largest datasets.

Datasets	Cholesterol		Fried	
Prot. selected	5	10	5	10
Method	**Runtime**			
Random	20s	28s	29min55s	41m14s
GA (GFS)	2m5s	2m30s	1h52min	2h50min
M-PEER (GFS+E)	33m	31m	2h28min	1h48min
M-PEER (S+F+E)	35m30s	26m	2h37min	1h51min

that the explanation provided by the M-PEER method are closer, i.e., they represent the evaluated instance with greater precision.

5.3.4 A note on execution time

This section discusses the execution time of the proposed methods. All experiments were run on a machine with the following configuration: Intel Xeon 2.2GHz, 16 CPUs, 125G RAM and Debian GNU/Linux 10 operating system. Prior to the evolutionary process, we loaded tabular data from the datasets into memory and computed the normalized values of local stability and local fidelity for each pair of instances of the training set, an algorithm with quadratic complexity. This is a costly process for large datasets, taking approximately 5hrs 49mins on the largest (fried) but 0.7 seconds on the smallest dataset (cholesterol). The execution time for running the algorithms after this preprocessing step, for the smallest and largest datasets, are reported in Table 5.8.

Finally, as reported in Section 5.2, the evolutionary algorithms were configured with different values for population size and number of generations, with 600 individuals in 150 generations for the single-objective GA and 300 individuals in 75 generations for the three versions of M-PEER. This was done to balance the computation time spent on each of the strategies, since the computational cost to evaluate multiple objectives is in general more expensive than evaluating a single objective. Further, the results shown that, with the current parameters, M-PEER results converge. As a consequence of this design decision, the single-objective GA is capable of evaluating a much larger number of solutions than the M-PEER versions, but still M-PEER obtains better overall results

than the GA.

5.4 Summary

This chapter proposed a new two-level strategy for explaining regression models and their predictions through the selection and assignment of prototypes. The explanation based on prototypes is able to select the most explanatory instances of the model and works in a similar way to the human reasoning of comparing similar cases previously seen to understand a new phenomenon. This methodology is instantiated with M-PEER, a multi-objective method based on SPEA2. We compared different versions of M-PEER, considering both the interpretability and the error of the model, with ProtoDash – considered state-of-the-art in this task, and other methods of correlated areas of machine learning We also adapted metrics proposed in the context of classification to evaluate the quality of prototypes in the context of regression. Furthermore, our methodology – unlike other approaches presented in the literature for explanation based on examples, – considers both the features that describe the instances and the model output that we want to explain. In this way, we are more faithfully explaining the model and not the initial dataset. A great advantage of this approach is that the model does not always fit perfectly the data but explains the model as faithfully as possible, given the user more arguments for a possible rejection or acceptance of the model for use in a given problem.

Our experiments demonstrated that M-PEER with two or three objectives (i.e., $GFS + E$ or $S + F + E$), together with a single GA based on RMSE, were equal or statistically superior to all other methods analyzed in terms of RMSE. Considering the GFS metric, all evolutionary methods – except the single-objective GA based on RMSE, obtained no difference or statistically better results than the competitors. With this we conclude that the M-PEER versions that take into account the explanatory characteristics of global fidelity, global stability and the error in the fitness minimization process, are the methods that best explain regression problems while simultaneously balancing all the criteria. Furthermore, we observed that the multi-objective version M-PEER $GFS, RMSE$ had a slight advantage over the version with three objectives when a larger number of prototypes is selected for RMSE metrics, possibly due to the greater simplicity imposed on the algorithm SPEA2 when minimizing only two objectives. However, note that the use of the evolutionary method is worthless when the dataset has a very small number of instances (here we considered 500 as a baseline for small datasets). In this case, the trade-off between computational cost and effectiveness is too low, making the use of the method not recommended.

It is important to note that the prototype-based explanation method proposed in this chapter, while similar to the way we often justify events based on other similar occurrences in our daily lives, does not fully explain why a similar instance resulted in a specific output. With this in mind, in the next chapter we propose a hybrid approach that not only justifies the model based on similar instances in the training set but also identifies the relevance of questions related to the features that determined the output, such as their importance and expected values in similar instances.

Chapter 6

HuMiE: A hybrid multilevel approach to explaining regression models

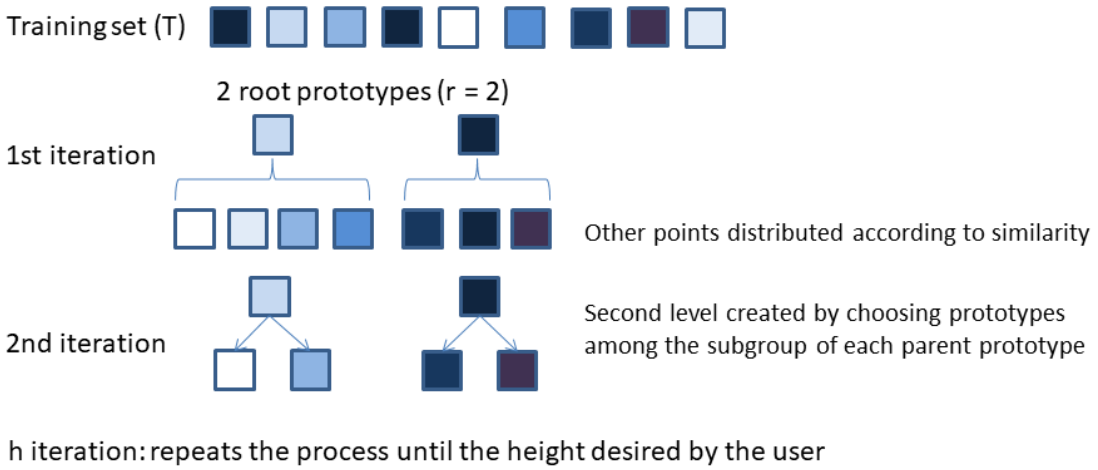
In this chapter we present a multilevel hybrid approach to explain black-box models in regression problems. We call this approach Hybrid Multilevel Explanation (HuMiE). By hybrid we mean that the explanation provided will have elements based on examples - through the selection of prototypes from the training set - and these examples will be associated with information from explanations based on features. The method gives emphasis to the most important characteristics of each prototype and their range of expected values.

As for the multilevel character, the proposed method provides explanations that present characteristics coming from different levels of the model: i) global explanations, with the most relevant prototypes; ii) local characteristics of test instances, showing how a specific test instance fits into the context of the problem as a whole; and iii) features at an intermediate level, portraying groups represented by prototypes capable of subdividing the problem into semantically similar partitions. This multilevel hybrid approach has a tree-like format.

To build the Hybrid Multilevel Explanation, HuMiE uses the two methods previously proposed, namely: DELA (Chapter 4) for the features-based explanation step, and M-PEER (Chapter 5) for the prototype selection step.

One of the few works that has already developed a similar approach was MAME (Model Agnostic Multilevel Explanations) [102]. As per MAME, little attention has been given to obtaining insights at an intermediate level of model explanations. MAME is a meta-method that builds a multilevel tree from a given method of local explanation. The tree is built by automatically grouping local explanations, without any hybrid character combining different types of explanation. The intermediate levels are based on this automatic grouping of the local explanations. Furthermore, the local explanation methods evaluated in MAME were not specifically built to work with regression problems, which can cause some damage when we start to have a continuous range of possible values for each instance output. Finally, MAME did not make the source code available and neither did they make available the tree generated for the only regression dataset used in their

Figure 6.1: Example of the multilevel tree built by HuMiE. In this figure, the squares represent instances, and the colors of the instances the feature values used to calculate similarities.



Source: created by the author.

work (Auto-MPG), which makes any kind of comparison between the approaches difficult.

The rest of this chapter is organized as follows. Section 6.1 introduces the proposed method. In section 6.2, we show the settings of the experiments carried out. In section 6.3, we conducted a quantitative and qualitative evaluation of the prototypes chosen by HuMiE, and performed a qualitative analysis of the classic Auto-MPG dataset. In section 6.4, we conducted a case study using responses from two commonly used questionnaires for assessing individuals' mental health as our dataset. Finally, section 6.5 draws conclusions.

6.1 Proposed Method

The section introduces Hybrid Multilevel Explanation (HuMiE), an explanation method that combines both example-based and feature-base explanations. HuMiE produces a tree where each node represents a prototype selected from the training set. After prototypes are selected, a feature-based approach is used to provide further details on the models decision.

The dynamics of creating this explanation tree is exemplified in Figure 6.1. In the figure, each square represents an instance from the training set (T), and the colors of the squares reflect the values of the features of the instances. We calculate the similarity between instances using these features. In this example, HuMiE first selects two prototypes as the root of the tree (r). After that, the distance from the remaining instances to each prototype is calculated, and each instance associated with its most similar prototype,

generating 2 subgroups. For each of these subgroups, again two children prototypes are chosen. This process is repeated until it reaches the height (h) of the tree defined by the user.

Having a tree of prototypes, for each node of the tree, HuMiE shows the most relevant features considering the prototype associated with that node, together with the range of values assumed by the instances of the subgroup represented by that prototype. Finally, HuMiE allows for a local explanation of a test instance by selecting the leaf-level prototype most similar to the instance to be explained.

By providing a hybrid explanation, HuMiE fills an important gap in explanation methods, as it can capture important features from different aspects. On the one hand, it can resemble the human way of explaining things, by indicating similar past events, while on the other hand, it can also explain why these past events are similar to the current scenario being evaluated. Moreover, in its multilevel character, HuMiE allows for an analysis capable of specializing global explanations into intermediate levels, representing semantically similar subgroups, and even into a locally coherent explanation that fits into the overall explanation.

The method is illustrated in Algorithm 6. It receives as input the training (T) and test (T') sets, together with the height h of the explanation tree to be built and the number r of root nodes present in the tree, where the root nodes represent the global explanations of the model.

In line 1, we create the regression model to be explained. Note that HuMiE is model agnostic, meaning that it has the ability to explain any regression model. To ensure semantically valid results, HuMiE works with continuous features only. Any categorical feature should be pre-processed. Next, we initialize some important variables for building the multilevel tree. In line 2, the tree is initialized as an array. This array will be filled with tuples containing three pieces of information: the selected prototype, the prototype's depth level in the tree, and the prototype at the previous level (parent) of the selected prototype. In line 3, a dictionary is initialized to store, for each selected prototype, the minimum and maximum values of each feature found in the subset of instances from where the prototype was chosen. This information will serve in the future to show the minimum and maximum values of each feature within the subset that the prototype will be representing. In line 4, an array is initialized to store the prototypes already chosen during the tree construction process. Finally, on line 5, we initialize an array to store the set of prototypes selected at the top level. As the explainer is built on a top-down fashion (i.e., starting from the root level), it has no higher-level prototype, and this variable is initialized with -1. In line 6 it is defined that the number of prototypes of the root level is the value of r received as a parameter in the algorithm.

After that, we carry out the selection of the prototypes that will belong to the tree in the loop between lines 7 and 29. Note that the construction of this tree will start from

Algorithm 6 Hybrid Multilevel Explanation (HuMiE)

Require: T (train), T' (test), h (height of explanation tree), r (number of tree root prototypes)

1: f = Regression model(T)
2: $tree$ = [ ]
3: $rangeAtt$ = {}
4: $selectedPR$ = [ ]
5: $prevLevelPR$ = [−1]
6: $numPR = r$
7: **for** each $depth \in$ range(h) **do**
8: $nearestPR$ = [ ]
9: **if** $depth > 0$ **then**
10: $numPR = 2$
11: **for** $t \in T$ **do**
12: $nearestPR$.append(selectPR(t, $prevLevelPR$, f))
13: **end for**
14: **end if**
15: **for** $previousPR \in prevLevelPR$ **do**
16: $validIndexes = (T \notin selectedPR) \wedge (T \in nearestPR(previousPR))$
17: **if** $depth == 0$ **then**
18: $validIndexes = T$
19: **end if**
20: $prototypes$ = MPEER($validIndexes$, $numPR$, f)
21: $tree$.append($prototypes$, $depth$, $previousPR$)
22: $rangeAtt[previousPR]$ = findMinMaxFeatures($validIndexes$)
23: $prevLevelPR = prototypes$
24: $selectedPR$.append($prototypes$)
25: **if** $depth = h$ **then**
26: $leaves = prototypes$
27: **end if**
28: **end for**
29: **end for**
30: $topFeatures$ = {}
31: **for** $PR \in selectedPR$ **do**
32: $topFeatures[PR]$ = DELA(PR)
33: **end for**
34: $localExplanation$ = {}
35: **for** $t \in T'$ **do**
36: $localExplanation[t]$ = selectPR(t, $leaves$, f)
37: $topFeatures[t]$ = DELA(t)
38: **end for**
39: drawHybridTree($tree$, $topFeatures$, $rangeAtt$, $LocalExplanation$)

the root, which will represent the global explanation of the model. After that, for each root level prototype selected, we expand it up to the level at limit h, defined by the user. In line 10 it is defined that for intermediate levels of the explanation tree we will have a

binary division of the prototypes. In the loop of lines 11 to 13, we check, for all instances of the training set, which prototype of the previous level is the most similar to represent it. We are redistributing the training points to identify the most suitable prototypes in a subset. Note that if we are at the first level this array will be an empty set (line 8). To measure the similarity between the instances we used the L2 of the normalized values of LF and LS measure (Equation 5.6).

In the loop between lines 15 and 28 we find the prototypes that represent each of the subsets divided by the prototypes in the previous iteration. To accomplish this, in line 20, we check the indexes of valid training points, i.e., all training points explained by the same prototype in the previous tree level, excluding instances already selected as prototypes. The conditional performed on line 17 verifies whether the chosen prototypes are from the root level, if so, all instances of the training set make up the list of valid indexes. In line 20, the M-PEER method is called for this subset of instances and the chosen prototypes are added to the tree in line 21. In line 22, the minimum and maximum values of all instances used to select each of the prototypes are checked, and in line 24 the set of selected prototypes is added to the array of previously selected prototypes. Finally, the condition in line 25 tests whether we have reached the last level h defined by the user, which corresponds to the leaf nodes of the tree that will be later associated with test instances.

The selection of the most relevant attributes and the range of expected values for each attribute of similar instances is performed in the loop between lines 31 and 33 for selected prototype points at all levels of the tree. The same procedure of feature analysis is performed in the loop between lines 35 and 38 for the set of test instances. This is done by DELA. In line 36, we check which leaf node is closest to the test point to associate the two in the final view of the tree.

Finally, on line 39, a function is called to draw the tree, showing the most relevant features and the range of expected features for the selected prototype instance and for the test instances.

6.2 Experimental Setup

We perform both a quantitative and a qualitative evaluation of the prototypes chosen by HuMiE and their characteristics. In the quantitative evaluation, we compare the prototypes chosen by HuMiE for local explanation with those chosen by other methods of explanation based on examples: ProtoDash and M-PEER. For that, we use the 25 datasets presented in Table 5.1. To measure the quality of the prototypes, we use the

classic RMSE (Eq. 5.1) and also the metrics of GF (Eq. 5.3), GS (Eq. 5.4) and GFS (Eq. 5.7) proposed earlier in Chapter 5. We also performed a qualitative analysis on the classic Auto-MPG dataset, previously used in Chapter 4, which portrays the fuel consumption of automobiles (miles per gallon – MPG) according to their construction characteristics. These results are reported in Section 6.3.

Finally, we performed a case study using 2 versions of 4 datasets collected by [78] from the answers to two questionnaires commonly used to assess the mental health of individuals, namely: WHOQOL-BREF[1] (World Health Organization Instrument to evaluate Quality of Life) and BSI [25] (Brief Symptoms Inventory). WHOQOL-BREF measures an individual's quality of life in different aspects. The BSI questionnaire measures symptoms related to conditions such as anxiety and depression. The case study is detailed in Section 6.4.

6.3 Experimental Analysis

This section presents first a quantitative experimental analysis of the explainability method proposed. The models to be explained were built using a Random Forest Regression algorithm, with implementation provided by sklearn[2]. All parameters were used with their default values.

Taking as input the model built using each of the datasets shown in Table 5.1, we run the explanation methods ProtoDash, M-PEER and HuMiE to compare the quality of the selected prototypes. For the ProtoDash explanation method we selected 6 global explanation prototypes and assigned each of the test instances to the most similar. For the M-PEER method we selected 3 global prototypes that are used for local explanation according to the proximity of the test instances. Finally, in the HuMiE method we select 3 global prototypes (the same as in the M-PEER method) to generate a sublevel of 6 prototypes that were then used to explain the test instances.

As previously said, we compared the results obtained in terms of the classic RMSE error metric to verify the ability of the model to be reconstructed only with the selected prototype instances. In addition, we use the metrics of Global Fidelity (GF), Global Stability (GF) and GFS to measure the semantic quality of the prototypes used for local explanation.

The results obtained are shown in Table 6.1, where the best ones are highlighted

[1] https://www.who.int/tools/whoqol

[2] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

Table 6.1: Result of the quality of the prototypes chosen by different methods in the local explanation of the test instances.

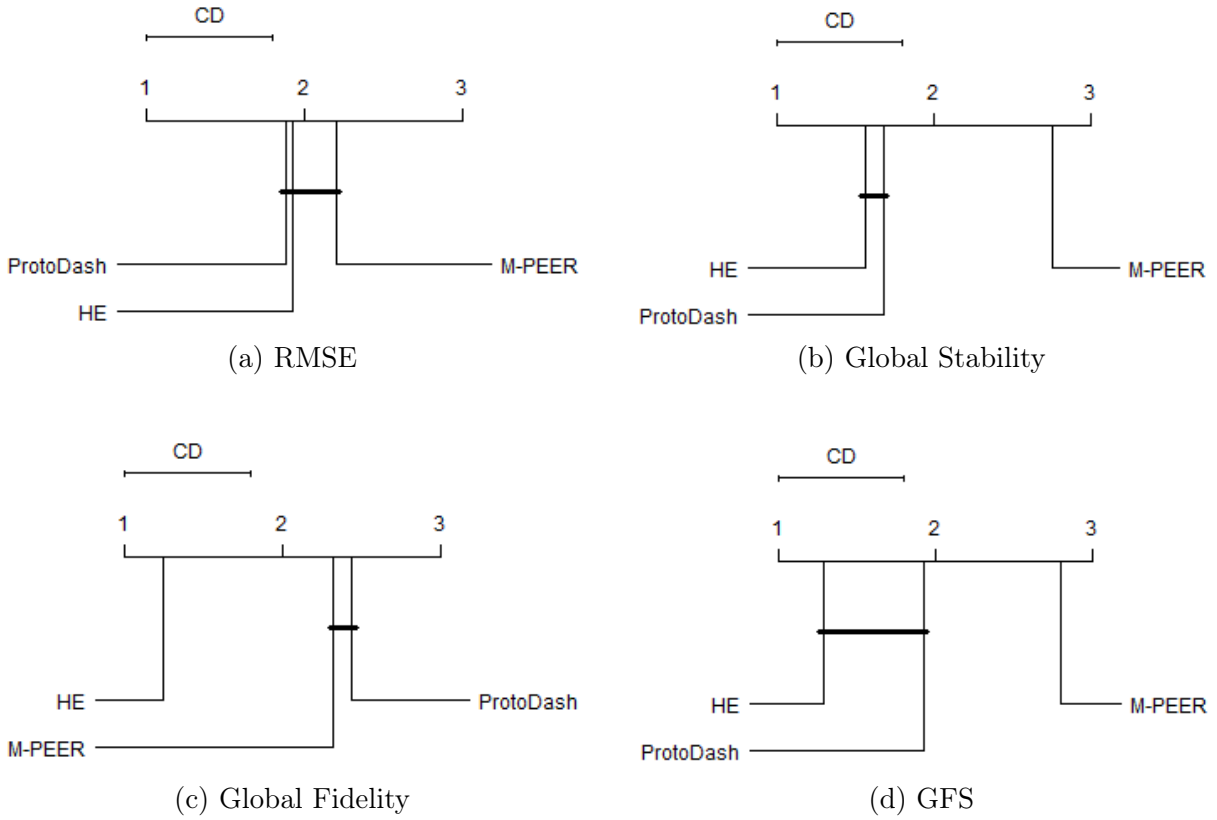
	ProtoDash				MPEER				HuMiE			
Dataset	GS	GF	GFS	RMSE	GS	GF	GFS	RMSE	GS	GF	GFS	RMSE
cholesterol	**0.06**	433.10	0.78	0.98	0.11	463.83	0.81	0.69	0.08	**354.21**	**0.75**	**0.61**
auto-mpg	**0.02**	117.60	0.60	**0.56**	0.07	96.17	0.63	0.81	0.05	**73.77**	**0.60**	0.68
sensory	0.14	478.57	0.85	1.26	0.19	465.81	0.86	**0.71**	**0.10**	**337.01**	**0.77**	0.72
strike	**0.02**	153.61	0.59	0.61	0.03	171.31	0.61	**0.53**	0.02	**118.43**	**0.57**	0.59
day_filter	0.11	409.93	0.82	0.85	0.09	608.41	0.88	0.73	**0.03**	**396.44**	**0.77**	**0.68**
qsar_fish	**0.04**	68.37	**0.56**	0.75	0.10	69.00	0.60	0.71	0.05	**59.99**	0.56	**0.60**
concrete	0.10	**282.32**	0.74	**0.77**	0.11	307.73	0.76	1.06	**0.06**	283.02	**0.72**	0.79
music	0.11	**1.86e+5**	**1.25**	**0.82**	0.13	1.97e+5	1.29	0.88	**0.09**	1.95e+5	1.27	0.99
house	0.05	9.58e+4	0.74	0.78	0.07	9.64e+4	0.75	0.84	**0.05**	**8.45e+4**	**0.72**	**0.68**
wineRed	**0.02**	1656.46	**0.43**	0.73	0.06	1391.73	0.48	0.68	0.06	**1230.08**	0.48	**0.60**
communities	**0.10**	2.08e+6	1.43	0.94	0.13	1.91e+6	1.39	0.66	0.10	**1.70e+6**	**1.30**	**0.62**
crimes	**0.00**	1.56e+6	1.16	**0.35**	**0.00**	1.66e+6	1.20	0.35	**0.00**	**1.54e+6**	**1.16**	0.36
abalone	0.05	525.57	0.42	0.93	0.06	386.61	0.43	0.70	**0.01**	**338.52**	**0.35**	**0.60**
wineWhite	**0.01**	1831.79	**0.40**	0.68	0.05	1544.18	0.45	**0.67**	0.05	**1415.63**	0.45	**0.66**
cpu_act	0.00	4972.93	0.40	**0.94**	0.01	**2978.10**	0.40	0.98	**0.00**	3076.94	**0.38**	1.25
bank32nh	**0.00**	7.05e+4	**0.53**	**0.81**	0.02	7.01e+4	0.55	0.93	0.05	**6.76e+4**	0.58	0.96
puma32H	**0.02**	**8.21e+4**	**0.57**	1.01	0.21	8.37e+4	0.66	**1.00**	0.05	8.51e+4	0.59	1.01
compactiv	0.01	4035.01	0.41	**0.91**	0.01	3106.38	0.40	0.97	**0.00**	**2945.77**	**0.38**	1.04
tic	**0.00**	1.13e+5	0.50	**0.72**	**0.00**	1.01e+5	0.49	**0.72**	**0.00**	**8.37e+4**	**0.47**	**0.72**
ailerons	0.02	2.40e+4	0.47	**0.75**	0.05	2.26e+4	0.49	0.95	**0.01**	**2.16e+4**	**0.46**	0.86
elevators	0.01	4277.07	0.38	**0.84**	0.02	**3706.47**	0.40	0.89	**0.01**	3763.58	**0.38**	0.88
california	0.02	670.03	0.42	0.91	0.07	478.54	0.48	1.08	**0.02**	**470.18**	**0.41**	**0.83**
house_16H	0.01	3627.19	0.37	**0.75**	0.00	2745.08	0.36	0.98	**0.00**	**2646.99**	**0.36**	0.81
fried	**0.28**	1253.83	**1.56**	**0.85**	0.39	1357.11	1.65	0.94	0.29	**1248.61**	**1.56**	0.97
mv	0.04	525.34	1.00	**0.81**	0.06	590.88	1.06	1.00	**0.02**	**504.68**	**0.98**	0.87

in bold. In addition, we show the critical diagrams of each of the metrics in the graphs in Figure 6.2. As stated earlier, these diagrams were generated after carrying out an adapted Friedman test followed by a Nemenyi post-hoc with a significance level of 0.05. As we are comparing multiple approaches, Bonferroni's correction was applied to all tests [24]. In these diagrams, the main line shows the average ranking of the methods, i.e., how well one method did when compared to the others. This ranking takes into account the absolute value obtained by each method according to the evaluated metric. The best methods are shown on the left (lowest rankings). The critical difference interval (CD in the plots) is determined by the Friedman test according to the significance level defined. If the average difference between two algorithms is greater than CD, then the null hypothesis that the algorithms have the same performance is rejected. Finally, the diagram connects groups of methods that do not present statistically significantly differences. Note that the size of the lines connecting the methods corresponds to the size of the CD interval.

As we can see, the creation of a prototype explanation sublevel managed to maintain the same qualities in terms of RMSE as its competitors, where we cannot say that it was statistically superior. Furthermore, the HuMiE approach was statistically better than the explanation with only one M-PEER prototype level in relation to Global Stability and GFS. Finally, the HuMiE multilevel hybrid approach was statistically better than all competitors in terms of the Global Fidelity of the chosen prototypes.

Next, we show a qualitative analysis for Auto-MPG, which has 398 instances described by 8 features. To evaluate the model error, we divided the data into two subsets,

Figure 6.2: Critical diagrams of the choice of prototypes for local explanation of test instances according to RMSE metrics (a), Global Stability (b), Global Fidelity (c) and GFS (d). From left to right, methods are ranked according to their performance, and those connected by a bold line present no statistical difference in their results.



(a) RMSE (b) Global Stability

(c) Global Fidelity (d) GFS

Source: created by the author.

the first being the training set with 70% of the instances (279) and the test set with the remaining 30% (119). Note that in this case the model error is used only to identify how good the explanation model is related to the original regression model to be explained. As our focus is the explanation provided and not minimizing the error, we do not perform any further evaluation, for example, through cross-validation.

The Auto-MPG dataset portrays a classic regression problem whose objective is to find the consumption of a car having as features constructive characteristics of the vehicle. To evaluate the explanation provided by our HuMiE approach we build a model using all available features in a Random Forest algorithm. The RMSE error found on the test set was 0.36.

Figure 6.3 shows the explanation provided for the model. In the explanations provided in this work we defined the height h of the tree as 2 and the total number of prototypes in the global explanation of the root as 3. Two test instances, one with a low output value and another with a high output value, were previously selected for local explanation.

In the explanation, each node highlighted in gray represents a prototype selected

Figure 6.3: Hybrid tree explanation for the Auto-MPG dataset.

Source: created by the author.

from the training set and the nodes, highlighted in red represent two test instances selected for local explanation. In addition, within each of the nodes, the actual output value of the instance (y) and the respective value predicted by the regressor model (y') are presented in the first line. The other lines of the tree node represent the 5 main features identified for that instance with their respective value (the most relevant number of features to be shown can easily be changed if the user wants to).

The bars represent the observed threshold values between all valid closest points when finding the prototype. In the case of the first upper level (root of the tree) the limit is imposed by the values of all instances of the training set. In the case of the test node the limit is imposed by all the closest features that originated the prototype local explanation leaf (leaf node of the hybrid explanation tree). The red dot represents the feature-specific value for the instance. The green range represents the allowable variation for creating a similar instance, determined by the DELA method. Specifically, it represents the range of feature values found among instances that deviate no more than 10% from the output predicted by the model. When hovering over one of the features, the green range is shown

numerically with a highlight balloon for the user.

Note that the global explanation prototypes managed to cover the solution space well, with the first prototype being a high fuel consumption car (14.5 MPG), the second an intermediate consumption car (19.08 MPG) and the last one a low fuel consumption car of fuel (28.05 MPG). In addition, among the most relevant features at this level, we find those that domain experts traditionally portray as determining the consumption of a car, we can highlight: weight, displacement, horsepower and acceleration.

In the second level (intermediate explanation) we observe that the prototypes really represent subgroups of the prototypes of the global explanation. This observation can be made both in terms of the output returned by the model and the range for each of the features. In the output we can observe that the values obtained do not extrapolate to values of the side prototypes of the previous level. In the range of features, we observed, for example, that in the first prototype of the second level the displacement feature varied between 260 and 455 while in its parent prototype these values varied between 68 and 455, portraying a specification of the group.

Finally, we can also observe that the L2FS similarity metric was adequate when associating the leaf prototypes of the explanation tree with the test instances, since both the values returned by the regressor model and the feature intervals represent the best possible association in each specific scenario.

6.4 Case Study: Explanining Models related to Mental Health

To build the models to be explained we used two regression algorithms, they are: i) Random Forest Regression; and ii) Linear Regression. We chose Linear Regression as one of the algorithms since the medical datasets used in this work have a linear combination of their features as output. Therefore, for this specific problem, we expect this method to provide the best possible results. All algorithms were run with their default parameters.

The datasets analyzed here were collected by [78] in order to assess how the coronavirus pandemic (SARS-CoV-2) affected people's mental health. To this end, measurements were performed using the World Health Organization Instrument to evaluate Quality of Life (WHOQOL-BREF) and Brief Symptoms Inventory (BSI). Both questionnaires are composed of questions that are answered on a 5-point Likert scale, ranging from 0 (not at all) to 4 (extremely). As an output, these tests result in a score per individual that is computed based on a linear combination of the answers provided. A complete break-

down of the WHOQOL-BREF and BSI instruments, including all questions, is available in Appendix C. These tests were performed on a total of 153,514 individuals in the year 2020.

The WHOQOL-BREF test consists of 26 questions that address different aspects of patients' quality of life. The questions are divided into four domains: physical health (wphys), psychological (wpsy), social relationships and the environment (wenv). In the physical domain, the questions evaluate pain, energy, sleep, mobility, and activities of daily living. In the psychological domain, the questions address self-esteem, positive and negative emotions, thinking, body image, and spirituality. In the social relationships domain, the questions evaluate social interaction, social support, and personal relationships. Finally, in the environment domain, the questions address the physical aspects of the environment, access to health services, safety, housing, finances, transportation, and leisure. We work with all domains except social relationships. Thus, we initially have three sets of data in the context of this test.

The BSI is a test that assesses the presence of psychological symptoms in patients. The test consists of 53 questions, which cover nine different domains: somatization, obsession-compulsion, interpersonal sensitivity, depression, anxiety, hostility, phobic anxiety, paranoid ideation, and psychoticism. The answers provided by patients can indicate the presence and severity of symptoms in each of these domains, allowing for a detailed evaluation of patients' psychological profile. Our focus will be on the somatization dimension (persistence of symptoms unexplained by medical causes). The somatization domain consists of 7 questions that assess symptoms such as headaches, dizziness, chest pain, and gastrointestinal problems. In this way, we initially have one more set of data from this test.

In addition, we used two versions of the WHOQOL-BREF and BSI datasets. In the first version, all the features corresponding to each test were used, plus demographic information about the individuals (gender, age in months, level of education, marital status and ethnicity). The categorical attributes went through a preprocessing phase were they were converted using one-hot encoding, resulting in 19 demographic features. In addition, we remove instances with missing data. At the end, we were left with 112,726 instances.

The second version was a dataset where we removed the demographic features, keeping only the answers given by the individuals during the questionnaire response. In this dataset, instances with missing data were also excluded, also resulting in a total of 112,726 instances.

At the end, we have 6 datasets from the WHOQOL-BREF questionnaire and 2 sets of data from the BSI test, half of them only with the responses to the questionnaires and half with added demographical information.

The number of instances for each of the datasets was an issue for our runs. Given

our hardware limitation and considering that the auxiliary methods (DELA and M-PEER) of the HuMiE hybrid construction demand a lot of memory consumption in the construction of similarity matrices, it was not possible to use the complete dataset. In this way, we randomly select 30,000 instances for our training sets and 30,000 instances for the test sets, for each of the datasets.

The four datasets on mental health output a score calculated from the aggregation of questions answered by patients through a linear function. Thus, initially we will use only the features present in the questionnaires and train a model with a linear regression algorithm. The results in terms of RMSE error in the test set were null for all four models evaluated, as expected.

In this way, we will analyze the explanations provided by the proposed HuMiE method and compare them with explanations that would be provided by the popular local explanation method LIME. For that, we previously selected two instances for local explanation, one with a low score and the other with a high score. In the provided explanations we will show the 5 most relevant features found by LIME to explain each of the test instances. For the explanation provided by HuMiE we define 3 root prototypes, 2 levels of explanation and at each node of the tree the 5 most relevant features.

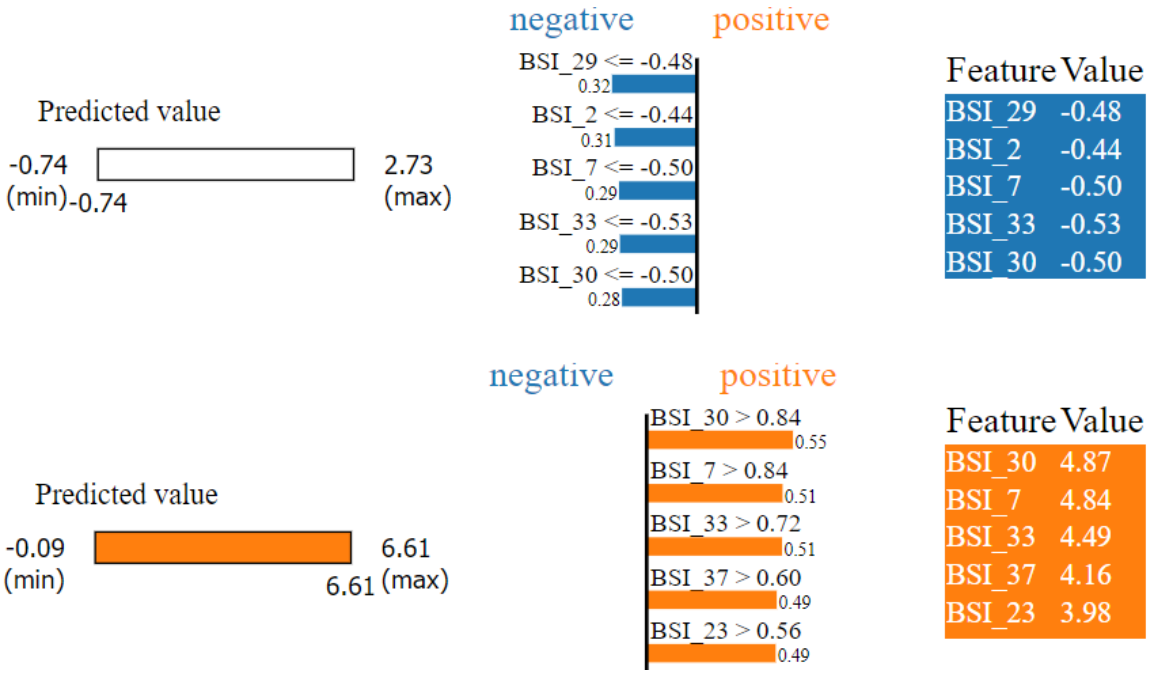
As previously mentioned, for the BSI questionnaire we focused our attention on the dimension of somatization as a model output. According to questionnaire, the final score obtained for a patient characterizes the intensity of suffering during the last seven days. Thus, the higher the score, the greater the suffering. Also according to the questionnaire, the features related to the somatization dimension are the questions: 2, 7, 23, 29, 30, 33, and 37. The output obtained by the linear regression model is shown in Equation 6.1. As we can see, only the specific features from the questionnaire obtained non-zero coefficients.

$$\begin{aligned}\widehat{f}(\mathbf{x}) = {} & 0.1682 \cdot BSI\_2 + 0.1967 \cdot BSI\_7 + 0.2302 \cdot BSI\_23 + 0.1948 \cdot BSI\_29 \\ & + 0.1957 \cdot BSI\_30 + 0.2088 \cdot BSI\_33 + 0.221 \cdot BSI\_37\end{aligned} \tag{6.1}$$

We can see in Figure 6.4, the LIME explanation method found the following features as most relevant for the low score output instance in the BSI questionnaire: 29, 2, 7, 33 and 30. As for the high score instance, the following features were found to be relevant: 30, 7, 33, 37 and 23. In addition to this information LIME also shows us the contribution of each feature to the calculation of the specific patient's score.

Figure 6.5 shows the explanation provided by HuMiE for the BSI questionnaire. For local explanation of the low score instance HuMiE chose the following features as most relevant: 37, 23, 33, 30 and 7. As for the high score instance, the chosen features were: 37, 29, 23, 30 and 7. In addition to this information HuMiE also shows locally which values are expected for these features in similar instances.

Figure 6.4: LIME Explanation for the BSI Dataset - Questionnaire features only.



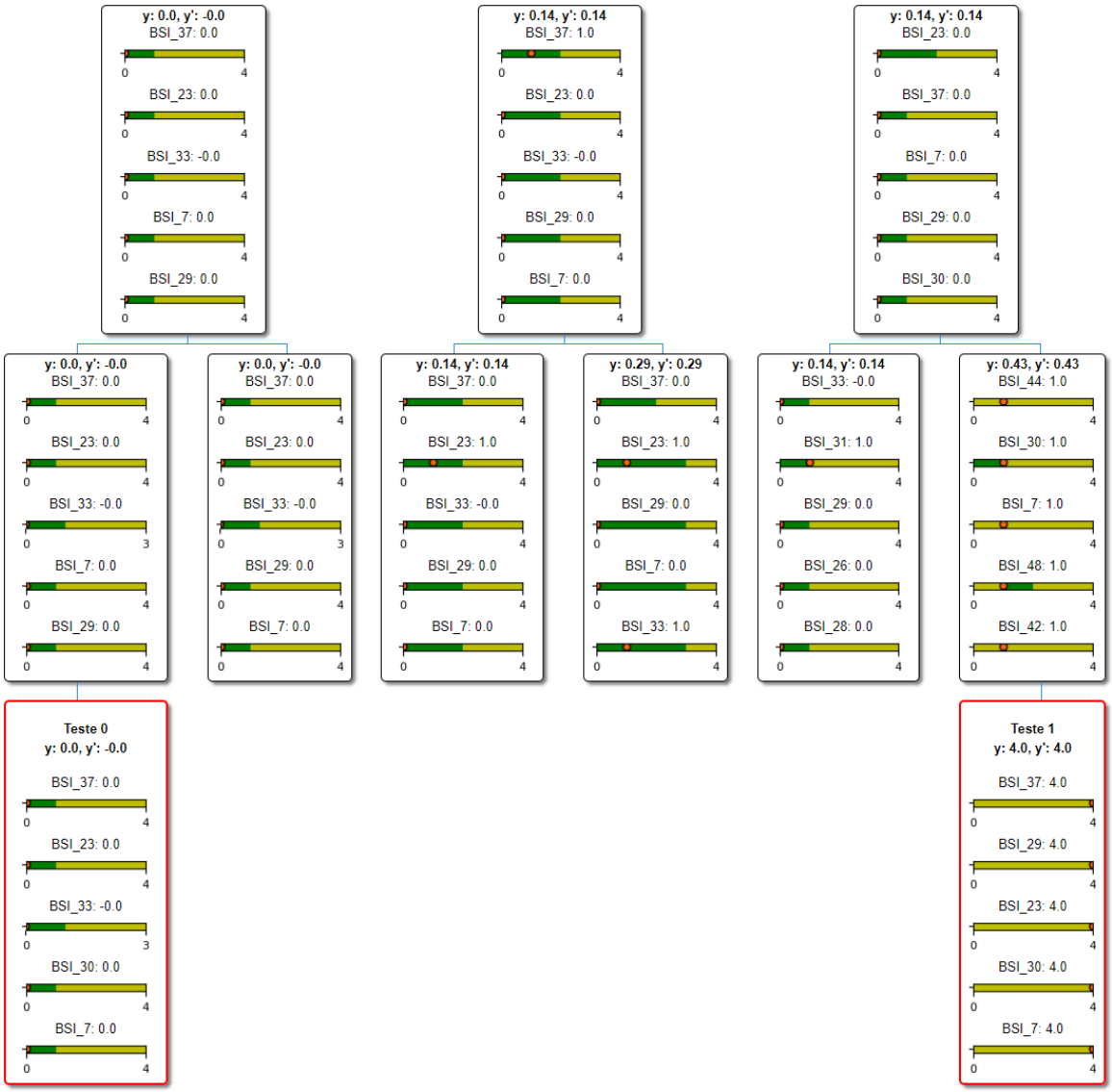
Source: created by the author.

Note that all sets of the 5 most relevant features found by both LIME and HuMiE are part of the features used to calculate the score for the BSI problem. We can highlight that the HuMiE explanation method also shows the global and intermediate explanation of the created model. In the global explanation HuMiE selected 3 prototypes. These root prototypes were binary divided into a second level of explanation. We can observe that the root level prototypes were not very discrepant in relation to the output obtained for the instances. This explanation is a direct result of the learning that was performed on the dataset, where few instances have high scores for the somatization dimension. In this dataset, approximately 90% of the instances have an output value less than or equal to 1. In this way, what at first glance could be considered a failure is actually portraying exactly the knowledge acquired by the model, after all it is not possible to learn something that is not contained in the data.

Finally, we can observe that with the multilevel explanation it also allowed access to other relevant features that are used in the calculation of the score (if the user did not know the function that originated it, it would be of great help to better understand the problem) and also to other features intermediaries that can be useful for a specialist to find hidden relationships in the data, for example, question 28 (Feeling afraid to travel on buses, subways, or trains) in patients with high scores.

As for the WHOQOL-BREF questionnaire, we will focus our attention on the wpsy domain. The other analyzed domains (wphys and wenv) presented similar results. In these tests, a lower score represents a worse quality of life for the patient in the analyzed domain. According to the questionnaire, the features that determine the wpsy domain

Figure 6.5: HuMiE Explanation for the BSI Dataset - Questionnaire features only.



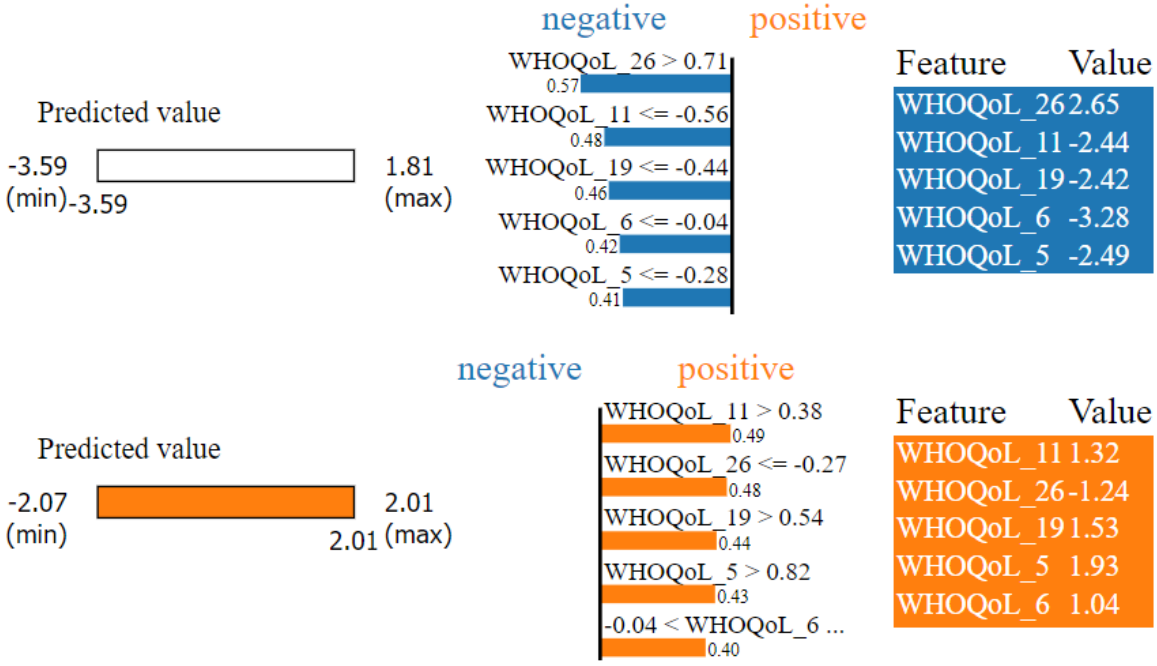
Source: created by the author.

are 5, 6, 7, 11, 19 and 26. The output obtained by the linear regression model is shown in Equation 6.2. As we can see, only the specific features from the questionnaire obtained non-zero coefficients.

$$\begin{aligned}\widehat{f}(\mathbf{x}) = 0.2113 \cdot WHOQoL\_5 + 0.2159 \cdot WHOQoL\_6 + 0.1965 \cdot WHOQoL\_7 \\ + 0.2478 \cdot WHOQoL\_11 + 0.2362 \cdot WHOQoL\_19 - 0.2403 * WHOQoL\_26\end{aligned} \tag{6.2}$$

The explanations provided by LIME are shown in Figure 6.6 and the explanations provided by HuMiE are shown in Figure 6.7. Again all 5 features found relevant by LIME are part of the set used to calculate the patient's score. In the explanation provided by HuMiE we find that all features used in the calculation are present at the root level and intermediate levels.

Figure 6.6: LIME Explanation for the WPSY Dataset - Questionnaire features only.



Source: created by the author.

Table 6.2: Error in regression models built with dataset including patient demographic features.

	RSME	
Dataset	**Linear Regression**	**Random Forest**
BSI	0.00	0.12
WHOQOL-BREF wpsy	0.00	0.08
WHOQOL-BREF wphys	0.00	0.10
WHOQOL-BREF wenv	0.00	0.12

In addition, in HuMiE for the local explanation of the instance with a low score, we also observed the presence of questions 1, 2 and 8. Questions 1 and 2 are general questions that directly inquire about the patient's perception of their quality of life and satisfaction with their health. They are not specific to any particular domain. Question 8 (How safe do you feel in your daily life?), despite not being used in the calculation of the specific domain, presents a similar context and, therefore, could also be used by the specialist to find hidden relationships in the data.

In future analyses, we included the demographic characteristics of patients in the datasets. In this way, we expect the problem to become more difficult. In this step, we used two algorithms to create the models: Random Forest and Linear Regression. The results of the errors in the test set are shown in Table 6.2. As we can see, the models created by Linear Regression kept errors at zero, being better than the Random Forest

Figure 6.7: HuMiE Explanation for the WPSY Dataset - Questionnaire features only.

Source: created by the author.

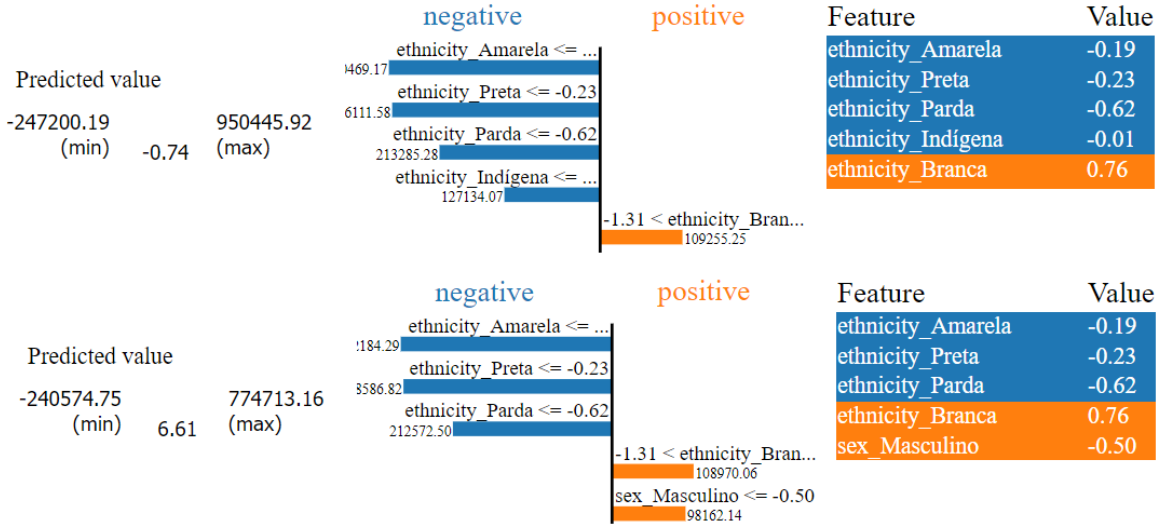
model in all datasets. In this way, we will continue to keep our attention on the linear model.

Explanations for the BSI datasets are shown in Figures 6.8 and 6.9 for the LIME and HuMiE explanation methods respectively. As we can see, the two methods of explanation listed among the most important features mostly demographic characteristics (HuMiE was able to identify some features related to the calculation of the score).

The result of the explanation could apparently be considered as an error, after all the calculated RMSE was zero. How is it possible that the score calculation features are not present? If we analyze the equation found by linear regression (Table 6.3), we observe that it corresponds to the same model created during the learning process.

Therefore, the explanations provided by our method are consistent with the model. Similar results were obtained for the other datasets used in this work.

Figure 6.8: LIME Explanation for the BSI Dataset - including features with demographic information about patients.



Source: created by the author.

Figure 6.9: HuMiE Explanation for the BSI Dataset - including features with demographic information about patients.

Source: created by the author.

Table 6.3: Equation obtained by linear regression using the BSI dataset containing demographic features.

$$
\begin{aligned}
\widehat{f}(\mathbf{x}) = \; & -48704.881 \cdot sex\_Feminino - 48704.881 \cdot sex\_Masculino \\
& -451.0402 \cdot Primarioincompleto \\
& -14390.4649 \cdot education\_Doutorado \\
& -1426.1002 \cdot PrimarioCompleto/Ginasioincompleto \\
& -2510.0293 \cdot GinasioCompleto/Colegialincompleto \\
& -21713.4741 \cdot education\_Mestrado \\
& -8000.589 \cdot Colegialcompleto/Superiorincompleto \\
& -26404.4766 \cdot education\_Superiorcompleto \\
& -24779.7538 \cdot marital\_status\_Casado(a)/Vivejunto \\
& -13498.5522 \cdot marital\_status\_Separado(a)/Divorciado(a) \\
& -24629.751 \cdot marital\_status\_Solteiro(a) \\
& +222829.0023 \cdot marital\_status\_Viuvo(a) \\
& +55079.3672 \cdot ethnicity\_Amarela \\
& +143103.371 \cdot ethnicity\_Branca \\
& +4197.7862 \cdot ethnicity\_Indigena \\
& +133172.2834 \cdot ethnicity\_Parda \\
& +66010.2356 \cdot ethnicity\_Preta \\
& +0.1682 \cdot BSI\_2 + 0.1967 \cdot BSI\_7 + 0.2302 \cdot BSI\_23 + 0.1948 \cdot BSI\_29 \\
& +0.1957 \cdot BSI\_30 + 0.2088 \cdot BSI\_33 + 0.221 \cdot BSI\_37
\end{aligned}
$$

6.5 Summary

This chapter introduced HuMiE, a method capable of creating multilevel hybrid explanations for regression problems. By hybrid we mean that the provided explanation encompasses as many elements of an explanation based on examples, through the selection of prototypes from the training set, as well as the presentation of explanations based on features such as, for example, the most relevant features and the expected values of features for the region of a given instance. By multilevel HuMiE presents in a tree format the model's global explanations, local explanations for specific test instances and also intermediate explanations, portraying semantically similar subgroups. Therefore, HuMiE uses the M-PEER approach as a submethod for choosing prototypes and the DELA approach for features analysis.

Experiments with real-world datasets quantitatively showed that the prototypes chosen by HuMiE were better than all competitors (M-PEER and ProtoDash) in relation to the fidelity metric and better than M-PEER with a single level in relation to the stability metrics and GFS. Qualitatively showed that HuMiE is able to diversify in the choice of prototypes according to characteristics of the presented dataset, both in terms of model output and features. In addition, HuMiE was able to find subgroups of instances that are

similar, providing an intermediate interpretation between the local and global scales. In this context, HuMiE was able to identify which features are most important according to the subgroup the instance belongs to. Finally, HuMiE also proved to be robust in terms of associating test instances with the best possibilities of tree leaf prototypes for local explanation.

Chapter 7

Conclusions and Future Work

This chapter draws the main conclusions about the results obtained in this thesis regarding the open issues defined in Chapter 1. We organize our results and conclusions according to these issues in the next sections, and then discuss directions of future work.

7.1 Issue 1: Regression vs Classification

Statement: In general, explanation methods were not designed and evaluated specifically for the context of regression.

As noted by the bibliographic study presented in Chapter 2, most of the methods proposed prior to the development of our work ignored inherent characteristics of the regression model to be explained in the explanation process. **This phenomenon occurred mainly in the proposals for explanation based on examples**.

To solve this problem, all our proposed methods take into account both the input of the instances described by the set of features as well as the continuous output provided as a preview for the instances. Thus, our explanations always aim to bring the understanding of the model and not the datasets.

Furthermore, all of our proposed approaches, regardless of feature-based (Chapters 3 and 4), example-based (Chapter 5) or hybrid (Chapter 6), have been exhaustively evaluated under specific datasets of regression problems, reaching a total of more than 20 datasets.

In the case of explanations based on examples, we proposed the approach of simultaneously minimizing multiple objectives, namely: i) based on features; ii) based on the output of the model; and iii) based on the error obtained by the explanation.

7.2 Issue 2: Local vs Global Explanations

Statement: The local explanation is not always coherent with the global explanation provided by the same explanation method.

We believe that the local and global explanation being coherent is fundamental for human understanding. In this way, our proposed approaches were based on the generalization and specification processes.

Generalization occurred in the process of feature-based explanations (Chapters 3 and 4). In this case, we initially obtained a set of local explanations for specific instances, and from the aggregation of these various explanations we built a coherent global explanation for several subspaces delimited by the model's output.

Specification took place in the process of example-based explanations (Chapter 5) and in the multilevel hybrid explanation (Chapter 6). In this case, we initially select a set of fixed size among the training instances to be the prototypes of the problem, representing the global explanation of the model. To specifically explain an instance maintaining process coherence, one of the local prototypes is selected, the one that maintains the greatest similarity in terms of a weighted equation that takes into account both the features of the problem and the output of the model.

7.3 Issue 3: Instances should be semantically valid

Statement: Semantically invalid instances for the application can be created by locally explaining the model.

Usually, algorithms that deal with local explanation create perturbations around samples of interest to generate fictitious samples and describe local behavior. However, this can lead to semantically invalid instances, which extrapolate possible values for input features or are simply not observed in the real world.

To avoid that, our approaches always use the model's own training points as a basis for explanations. We consider that if a model was built with such data, it must contain enough information to explain it. Thus, in feature-based explanations, our neighborhood region where linear regressions capable of explaining the model will be determined are constituted by previously selected training set points. In the explanations based on examples, the prototype points are also selected from the training dataset.

Although our methods do not rely on fictitious instances to construct explanations,

they are limited in situations where we do not have access to the training set used to generate the model being explained. Consequently, our methods may not be suitable for models created using private or confidential data.

7.4 Issue 4: Measuring semantics

Statement: Metrics to measure the similarity of instances and to evaluate the explanatory power of methods are little explored, and significantly impact the results of explanations.

We observe by studying the literature that, in most explanation methods, there is no proper care about which similarity metric will be used in internal processes where comparisons between different instances are necessary. For the most part of the studies, a simple Euclidean distance is used.

As we know, as in any machine learning problem that involves clustering, the similarity metric used is of paramount importance, and the inadequate use of distance measures may harm the problem. For example, we may found problems derived from the structure that represents the instances (real, natural numbers, classes, etc) or by the number of features of the problem (which can fall under the curse of dimensionality) .

To solve this problem in explanation methods for regression models, we evaluated the relevance of 8 different distance measures (Section 4.1) and found that there is no universally superior measure as the best one varies depending on the dataset in question. Therefore, we proposed an algorithm to select the best distance measure based on characteristics found in specific datasets. Our distance measure selection algorithm starts with the assumption that instances with similar output values should have relatively similar predictive features. Thus, the proposed distance selection method assumes the existence of linearity between the model's output for each instance and the distances that separate these instances.

To measure the quality of explanation, we also observed that in quantitative studies, many previous works used only error-based metrics (Accuracy, RMSE, etc.). Therefore, in addition to these metrics, we also proposed in the explanation processes based on examples the use of a metric that takes into account aspects that are semantically more understandable for a human, which are fidelity (related to the discrepancy between the model's output in relation to the explanation) and stability (related to the variability present in the set of features of an explanation in relation to the instance that is intended to be explained).

7.5 Future works

In this thesis, we focused on developing specific explanation methods for regression problems. However, we faced a significant challenge in comparing different explanation types quantitatively, such as feature-based, prototype-based, counterfactual, and visual explanations, as each method explains the model on a different aspect. We consider these methods to be complementary rather than competitors in understanding the model. However, it would be interesting to compare the level of understanding that each of these methods provides through automated means, without requiring human involvement. This would increase the scale of what could be evaluated, reduce the time spent on data collection, and remove individual subjectivity during the evaluation.

Another potential area for future research would be to investigate the applicability of our proposed methods in other types of regression problems, such as time-series and multi-output regression. Another broad area for future research is the interpretability of ensemble models using our proposed explanation methods. Ensemble models have become increasingly popular due to their ability to improve predictive performance, but their interpretability remains a challenge. By applying our explanation methods to ensemble models, we could gain insights into the contributions of individual models within the ensemble and the interactions between them. Additionally, developing new explanation methods that incorporate additional information, such as domain knowledge, into the explanation process would be a promising research direction.

Regarding our feature-based explanation method (DELA), we observed that a promising study would be to evaluate the variance of the importance of features in the local explanation. This deviation could portray the degree of uncertainty of the explanation and help the user determine the level of reliability they could have in a given region of space. Thus, this mechanism could serve as a subsidy for the user to improve the model in specific regions, such as locally increasing the number of training instances or adjusting the model parameters.

Moreover, it would be interesting to investigate the effectiveness of combining various types of explanations to improve overall model interpretability. For example, we combined feature-based and prototype-based explanations in the HuMiE method. Further exploration of these combinations could significantly advance the field of model explanation and increase the practical applicability of regression models in various domains.

Another potential study could evaluate the level of persuasion of the different approaches to check whether a user is likely to believe in a given explanation approach, even if the results do not match the trained model. This research direction could shed light on the human factor in model explanation, which is often overlooked.

Finally, we could use the proposed methods to verify the presence of bias in models.

As our explanations are obtained using the training set, we would have a direct correspondence if the model presents criteria related to injustice with the inputs that served to train the model. These future directions have the potential to significantly contribute to the field of model explanation and broaden its practical applicability.

Bibliography

[1] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.

[2] C.H. Achen. *Interpreting and Using Regression*. Number Nº 29 in 07. SAGE Publications, 1982.

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

[4] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. In Jan Van den Bussche and Victor Vianu, editors, *Database Theory — ICDT 2001*, pages 420–434, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[5] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional spaces. In *ICDT'01y*, pages 420–434, 2001.

[6] Lun Ai, Stephen H. Muggleton, Céline Hocquette, Mark Gromowski, and Ute Schmid. Beneficial and harmful explanatory machine learning. *Machine Learning*, 110(4):695–721, March 2021.

[7] Ahmed M. Alaa and Mihaela van der Schaar. Demystifying black-box models with symbolic metamodels. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[8] Jesús Alcalá-Fdez, Alberto Fernández, Julián Luengo, Joaquín Derrac, Salvador García, Luciano Sánchez, and Francisco Herrera. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing*, 17, 2011.

[9] Guilherme Seidyo Imai Aldeia and Fabrício Olivetti de França. Interpretability in symbolic regression: a benchmark of explanatory methods using the feynman data set. *Genetic Programming and Evolvable Machines*, May 2022.

[10] Guilherme Seidyo Imai Aldeia and Fabrício Olivetti de França. *Measuring Feature Importance of Symbolic Regression Models Using Partial Effects*, page 750–758. Association for Computing Machinery, New York, NY, USA, 2021.

[11] Daniel W. Apley and Jingyu Zhu. Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society Series B*, 82(4):1059–1086, September 2020.

[12] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58:82–115, jun 2020.

[13] Vijay Arya, Rachel K. E. Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Q. Vera Liao, Ronny Luss, Aleksandra Mojsilović, Sami Mourad, Pablo Pedemonte, Ramya Raghavendra, John Richards, Prasanna Sattigeri, Karthikeyan Shanmugam, Moninder Singh, Kush R. Varshney, Dennis Wei, and Yunfeng Zhang. One explanation does not fit all: A toolkit and taxonomy of ai explainability techniques. https://arxiv.org/abs/1909.03012, September 2019.

[14] Jacob Bien and Robert Tibshirani. Prototype selection for interpretable classification. *The Annals of Applied Statistics*, 5(4):2403–2424, Dec 2011.

[15] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[16] Nadia Burkart and Marco F. Huber. A survey on the explainability of supervised machine learning. *Journal of Artificial Intelligence Research*, 70:245–317, Jan 2021.

[17] Moritz Böhle, Mario Fritz, and Bernt Schiele. B-cos networks: Alignment is all we need for interpretability. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10319–10328, 2022.

[18] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, Jul 2019.

[19] Gary K. Chen, Eric C. Chi, John Michael O. Ranola, and Kenneth Lange. Convex clustering: An attractive alternative to hierarchical clustering. *PLOS Computational Biology*, 11(5):1–31, 05 2015.

[20] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. Modeling wine preferences by data mining from physicochemical properties. *Decis. Support Syst.*, 47(4):547–553, November 2009.

[21] Ian Covert, Scott Lundberg, and Su-In Lee. Understanding global feature contributions with additive importance measures, 2020.

[22] Dean De Cock. Ames, iowa: Alternative to the boston housing data as an end of semester regression project. *Journal of Statistics Education*, 19, 11 2011.

[23] Renato de Pontes Pereira. ARFF datasets. https://github.com/renatopp/arff-datasets, 2012.

[24] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, December 2006.

[25] LR Derogatis. Brief symptom inventory (bsi) administration, scoring, and procedures manual: Ncs pearson. *Inc., Minneapolis*, 1993.

[26] Daniel Deutch and Nave Frost. Cec: Constraints based explanation for classifications. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, pages 1879–1882, New York, NY, USA, 2018. ACM.

[27] Daniel Deutch and Nave Frost. Cec: Constraints based explanation for classifications. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 1879–1882, New York, NY, USA, 2018. Association for Computing Machinery.

[28] Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository. https://archive.ics.uci.edu/ml/, 2017.

[29] Pedro Domingos. A few useful things to know about machine learning. *Commun. ACM*, 55(10):78–87, October 2012.

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

[31] Finale Doshi-Velez and Been Kim. *Considerations for Evaluation and Generalization in Interpretable Machine Learning*, pages 3–17. Springer International Publishing, Cham, 2018.

[32] Pedro G Espejo, Sebastián Ventura, and Francisco Herrera. A survey on the application of genetic programming to classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40(2):121–144, 2010.

[33] Robert H. Essenhigh, H. Eugene Shull, Thomas Blackadar, and Herbert McKinstry. Effect of vehicle size and engine displacement on automobile fuel consumption. *Transportation Research Part A: General*, 13(3):175–177, 1979.

[34] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[35] Leonardo Augusto Ferreira, Frederico Gadelha Guimaraes, and Rodrigo Silva. Applying genetic programming to improve interpretability in machine learning models. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, New York, jul 2020. IEEE.

[36] Renato Miranda Filho, Anisio Lacerda, and Gisele L. Pappa. Explaining symbolic regression predictions. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, New York, jul 2020. IEEE.

[37] Félix-Antoine Fortin. et al. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, jul 2012.

[38] Hiroshi Fukui, Tsubasa Hirakawa, Takayoshi Yamashita, and Hironobu Fujiyoshi. Attention branch network: Learning of attention mechanism for visual explanation. In *ICPR'19*, 2019.

[39] Glenn Fung, Sathyakama Sandilya, and R. Bharat Rao. Rule extraction from linear support vector machines. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, page 32–40, New York, NY, USA, 2005. Association for Computing Machinery.

[40] Marzyeh Ghassemi, Luke Oakden-Rayner, and Andrew L Beam. The false hope of current approaches to explainable artificial intelligence in health care. *The Lancet Digital Health*, 3(11):e745–e750, November 2021.

[41] Bishwamittra Ghosh and Kuldeep S Meel. Imli: an incremental framework for maxsat-based learning of interpretable classification rules. In *AAAI/AIES*, pages 203–210, 2019.

[42] Raffaele Giancarlo, Giosuè Lo Bosco, and Luca Pinello. Distance functions, clustering algorithms and microarray data analysis. In *Proceedings of the 4th International Conference on Learning and Intelligent Optimization*, LION'10, pages 125–138, Berlin, Heidelberg, 2010. Springer-Verlag.

[43] PETER. GODDEN et al. Wine bottle closures: physical characteristics and effect on composition and sensory properties of a semillon wine 1. performance up to 20 months post-bottling. *Australian Journal of Grape and Wine Research*, 7(2):64–105, 2001.

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

[45] Vincent Grari, Sylvain Lamprier, and Marcin Detyniecki. Adversarial learning for counterfactual fairness. *Machine Learning*, pages 1–23, 2022.

[46] Ulrike Grömping. Variable importance in regression models. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(2):137–152, 2015.

[47] Andreia P. Guerreiro, Carlos M. Fonseca, and Luís Paquete. The hypervolume indicator: Problems and algorithms, 2020.

[48] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. Local rule-based explanations of black box decision systems, 2018.

[49] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM Comput. Surv.*, 51(5):93:1–93:42, August 2018.

[50] Rajeev Gupta, Madhawi Sharma, Neeraj Krishna Goyal, Preeti Bansal, Sailesh Lodha, and Krishna Kumar Sharma. Gender differences in 7 years trends in cholesterol lipoproteins and lipids in india: Insights from a hospital database. *Indian Journal of Endocrinology and Metabolism*, 20(2):211, 2016.

[51] Karthik S. Gurumoorthy, Amit Dhurandhar, Guillermo A. Cecchi, and Charu C. Aggarwal. Efficient data representation by selecting prototypes with importance weights. *ICDM'19*, 2019.

[52] Peter Hase, Chaofan Chen, Oscar Li, and Cynthia Rudin. Interpretable image recognition with hierarchical prototypes. In *AAAI'19*, pages 32–40, 2019.

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

[54] H. E. T. Holgersson, T. Norman, and S. Tavassoli. In the quest for economic significance: assessing variable importance through mean value decomposition. *Applied Economics Letters*, 21(8):545–549, 2014.

[55] Giles Hooker and Lucas Mentch. Please stop permuting features: An explanation and alternatives, 2019.

[56] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural*

Information Processing Systems, volume 32, pages 9737–9748, Red Hook, NY, USA, 2019. Curran Associates, Inc.

[57] Weronika Hryniewska, Adrianna Grudzień, and Przemysław Biecek. Limecraft: handcrafted superpixel selection and inspection for visual explanations. *Machine Learning*, pages 1–18, 2022.

[58] Texas Heart Institute. Cholesterol. https://www.texasheart.org/heart-health/heart-information-center/topics/cholesterol/, 2019. Access date: 08 August 2019.

[59] Aleks Jakulin, Martin Možina, Janez Demšar, Ivan Bratko, and Blaž Zupan. Nomograms for visualizing support vector machines. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, page 108–117, New York, NY, USA, 2005. Association for Computing Machinery.

[60] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014.

[61] Pablo A. Jaskowiak, Ricardo JGB Campello, and Ivan G. Costa. On the selection of appropriate distances for gene expression data clustering. *BMC Bioinformatics*, 15(2):S2, Jan 2014.

[62] Yunzhe Jia, James Bailey, Kotagiri Ramamohanarao, Christopher Leckie, and Michael E. Houle. Improving the quality of explanations with local embedding perturbations. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '19, page 875–884, New York, NY, USA, 2019. Association for Computing Machinery.

[63] Ying Jin, Weilin Fu, Jian Kang, Jiadong Guo, and Jian Guo. Bayesian symbolic regression, 2020.

[64] Margot E Kaminski. The right to explanation, explained. *Berkeley Tech. LJ*, 34:189, 2019.

[65] Jian Kang, Scott Freitas, Haichao Yu, Yinglong Xia, Nan Cao, and Hanghang Tong. X-rank: Explainable ranking in complex multi-layered networks. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, pages 1959–1962, New York, NY, USA, 2018. ACM.

[66] Andrei Kapishnikov, Tolga Bolukbasi, Fernanda Viegas, and Michael Terry. Xrai: Better attributions through regions. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4947–4956, 2019.

[67] Maarten Keijzer. Improving symbolic regression with interval arithmetic and linear scaling. In *Proceedings of the 6th European Conference on Genetic Programming*, EuroGP'03, pages 70–82, Berlin, Heidelberg, 2003. Springer-Verlag.

[68] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[69] Mirosław Kordos and Marcin Blachnik. Instance selection with neural networks for regression problems. In *ICANN'12*, pages 263–270. Springer, 2012.

[70] William La Cava, Patryk Orzechowski, Bogdan Burlacu, Fabricio de Franca, Marco Virgolin, Ying Jin, Michael Kommenda, and Jason Moore. Contemporary symbolic regression methods and their relative performance. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

[71] Thai Le, Suhang Wang, and Dongwon Lee. Grace: Generating concise and informative contrastive sample to explain neural network model's prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '20, page 238–248, New York, NY, USA, 2020. Association for Computing Machinery.

[72] Jonathan Lear et al. *Aristotle: The Desire to Understand*. Cambridge University Press, 1988.

[73] Simon Letzgus, Patrick Wagner, Jonas Lederer, Wojciech Samek, Klaus-Robert Muller, and Gregoire Montavon. Toward explainable artificial intelligence for regression models: A methodological perspective. *IEEE Signal Processing Magazine*, 39(4):40–58, 2022.

[74] Thomas Low, Christian Borgelt, Sebastian Stober, and Andreas Nürnberger. *The Hubness Phenomenon: Fact or Artifact?*, pages 267–278. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[75] O. Loyola-González. Black-box vs. white-box: Understanding their advantages and weaknesses from a practical point of view. *IEEE Access*, 7:154096–154113, 2019.

[76] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4768–4777, Red Hook, NY, USA, 2017. Curran Associates Inc.

[77] Cui Lv and Di-Rong Chen. Interpretable functional logistic regression. In *Proceedings of the 2nd International Conference on Computer Science and Application Engineering*, CSAE '18, New York, NY, USA, 2018. Association for Computing Machinery.

[78] Guilherme F. Marchezini, Anisio M. Lacerda, Gisele L. Pappa, Wagner Meira, Debora Miranda, Marco A. Romano-Silva, Danielle S. Costa, and Leandro Malloy Diniz. Counterfactual inference with latent variable and its application in mental health care. *Data Min. Knowl. Discov.*, 36(2):811–840, mar 2022.

[79] Ričards Marcinkevičs and Julia E. Vogt. Interpretability and Explainability: A Machine Learning Zoo Mini-tour. *arXiv preprint arXiv:2012.01805*, pages 1–24, 2020.

[80] D. Martens, B. B. Baesens, and T. Van Gestel. Decompositional rule extraction from support vector machines by active learning. *IEEE Transactions on Knowledge and Data Engineering*, 21(2):178–191, 2009.

[81] Trent McConaghy and Georges Gielen. Canonical form functions as a simple means for genetic programming to evolve human-interpretable functions. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 855–862. ACM, 2006.

[82] John Alexander McDermid, Yan Jia, Zoe Porter, and Ibrahim Habli. Ai explainability: The technical and ethical dimensions. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, 2021.

[83] David Alvarez Melis and Tommi Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Neurips'18*, pages 7775–7784, 2018.

[84] George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 101(2):343, 1994.

[85] Yao Ming, Panpan Xu, Huamin Qu, and Liu Ren. Interpretable and steerable sequence learning via prototypes. In *KDD'19*, page 903–913, 2019.

[86] L. F. Miranda, V. B. O. Luiz Otavio, B. S. M. Joao Francisco, and G. L. Pappa. Instance selection for geometric semantic genetic programming. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2020.

[87] R. Miranda Filho, A. Lacerda, and G. L. Pappa. Explaining symbolic regression predictions. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2020.

[88] Christoph Molnar. *Interpretable Machine Learning*. Lulu. com, 2 edition, 2022.

[89] Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 607–617, New York, NY, USA, 2020. Association for Computing Machinery.

[90] Martin Možina, Janez Demšar, Michael Kattan, and Blaž Zupan. Nomograms for visualization of naive bayesian classifier. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD '04, page 337–348, Berlin, Heidelberg, 2004. Springer-Verlag.

[91] Lori Newell. Cholesterol & your heart rate. https://www.livestrong.com/article/458203-cholesterol-your-heart-rate/, 2019. Access date: 08 August 2019.

[92] Ji Ni, Russ H. Drieberg, and Peter I. Rockett. The use of an analytic quotient operator in genetic programming. *IEEE Transactions on Evolutionary Computation*, 17(1):146–152, 2013.

[93] Minister of Natural Resources Canada. Learn the facts: Weight affects fuel consumption. *Auto$mart*, 2014.

[94] J. Arturo Olvera-López, J. Ariel Carrasco-Ochoa, J. Francisco Martínez-Trinidad, and Josef Kittler. A review of instance selection methods. *Artif. Intell. Rev.*, 34(2):133–143, August 2010.

[95] Patryk Orzechowski, William La Cava, and Jason H. Moore. Where are we now?: A large benchmark study of recent symbolic regression methods. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 1183–1190, New York, NY, USA, 2018. ACM.

[96] Eliana Pastor and Elena Baralis. Explaining black box models by means of local rules. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, SAC '19, pages 510–517, New York, NY, USA, 2019. ACM.

[97] Eliana Pastor and Elena Baralis. Explaining black box models by means of local rules. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, SAC '19, page 510–517, New York, NY, USA, 2019. Association for Computing Machinery.

[98] Gregory Plumb, Maruan Al-Shedivat, Eric P. Xing, and Ameet Talwalkar. Regularizing black-box models for improved interpretability. *CoRR*, abs/1902.06787, 2019.

[99] Hugo M Proença and Matthijs van Leeuwen. Interpretable multiclass classification by mdl-based rule lists. *Information Sciences*, 512:1372–1393, 2020.

[100] Nadia Rahmah and Imas Sitanggang. Determination of optimal epsilon (eps) value on dbscan algorithm to clustering data on peatland hotspots in sumatra. *IOP Conference Series: Earth and Environmental Science*, 31:012012, 01 2016.

[101] Karthikeyan Natesan Ramamurthy, Bhanukiran Vinzamuri, Yunfeng Zhang, and Amit Dhurandhar. Model agnostic multilevel explanations. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[102] Karthikeyan Natesan Ramamurthy, Bhanukiran Vinzamuri, Yunfeng Zhang, and Amit Dhurandhar. Model agnostic multilevel explanations. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[103] P. Rasouli and I. C. Yu. Explan: Explaining black-box classifiers using adaptive neighborhood generation. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9, 2020.

[104] Alexander Renkl. Toward an instructionally oriented theory of example-based learning. *Cognitive science*, 38(1):1–37, 2014.

[105] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386*, 2016.

[106] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should i trust you?" Explaining the predictions of any classifier. In *KDD'16*, pages 1135–1144, 2016.

[107] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144, New York, NY, USA, 2016. ACM.

[108] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *AAAI'18*, 2018.

[109] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

[110] Jakub Sliwinski, Martin Strobel, and Yair Zick. Axiomatic characterization of data-driven influence measures for classification. In *AAAI'19*, volume 33, pages 718–725, 2019.

[111] Yunsheng Song, Jiye Liang, Jing Lu, and Xingwang Zhao. An efficient instance selection algorithm for k nearest neighbor regression. *Neurocomputing*, 251:26 – 34, 2017.

[112] Erik Strumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. *J. Mach. Learn. Res.*, 11:1–18, March 2010.

[113] Xiangyu Sun, Jack Davis, Oliver Schulte, and Guiliang Liu. Cracking the black box. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Jul 2020.

[114] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[115] D. Roland Thomas, Edward Hughes, and Bruno D. Zumbo. On variable importance in linear regression. *Social Indicators Research*, 45(1):253–275, Nov 1998.

[116] Rebecca Wexler. When a computer program keeps you in jail. https://www.nytimes.com/2017/06/13/opinion/how-computers-are-harming-criminal-justice.html, 2017. Accessed: 2020-03-31.

[117] Michael Weylandt, John Nagorski, and Genevera I. Allen. Dynamic visualization and fast computation for convex clustering via algorithmic regularization. *Journal of Computational and Graphical Statistics*, 29(1):87–96, Jul 2019.

[118] Z. F. Wu, J. Li, M. Y. Cai, Y. Lin, and W. J. Zhang. On membership of black-box or white-box of artificial neural network models. In *2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*, pages 1400–1404, June 2016.

[119] Mengjiao Yang and Been Kim. Benchmarking attribution methods with relative feature importance, 2019.

[120] John Zerilli, Alistair Knott, James MacLaurin, and Colin Gavaghan. Algorithmic decision-making and the control problem. *Minds Mach.*, 29(4):555–578, 2019.

[121] Qingyuan Zhao and Trevor Hastie. Causal interpretations of black-box models. *Journal of Business & Economic Statistics*, 39(1):272–281, 2021.

[122] Zhengze Zhou, Giles Hooker, and Fei Wang. S-lime: Stabilized-lime for model explanation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '21. Association for Computing Machinery, 2021.

[123] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

[124] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. Spea2: Improving the strength pareto evolutionary algorithm. *TIK-report*, 103, 2001.

[125] Eckart Zitzler and Lothar Thiele. Multiobjective optimization using evolutionary algorithms — a comparative case study. In Agoston E. Eiben, Thomas Bäck, Marc Schoenauer, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature — PPSN V*, pages 292–301, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[126] Álvar Arnaiz-González, Marcin Blachnik, Miroslaw Kordos, and César García-Osorio. Fusion of instance selection methods in regression tasks. *Information Fusion*, 30:69 – 79, 2016.

Appendix A

Contracting different explanation methods

The main objective of this appendix is to show what kind of information we can extract from explanations provided by different methods and discuss the correct interpretation of the explanations. For that, we have selected two data sets and set of methods to evaluate. This section describes both and explains how experiments were designed, and reports the error rates obtained by different methods.

A.1 Datasets

As the objective is to perform an in in-depth comparison of explanations provided by different methods, we use two datasets as case studies, the former made of synthetic data, and the later with real-world data.

We used the Gaussian probability density function for the synthetic data set, with the generator function shown in Eq. A.1.

$$f(\theta, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\theta^2/2\sigma^2}, \tag{A.1}$$

where θ is the value of interest, and σ is the standard-deviation. This function assumes a mean 0 and represents the probability of observing a value $\theta - \Delta < \theta < \theta + \Delta$ for an infinitesimal Δ.

For the real-world example, we used the *Auto-MPG* data set from the UCI machine learning repository [28], describing the relationship between fuel consumption of a car (measured in miles per gallon - mpg) with seven features: cylinder ($Cyl.$), engine displacement ($Disp.$), horsepower (HP), weight ($Weight$), acceleration ($Acc.$), year of the model ($Model\ year$), and country of origin. The country of origin is the only categorical feature and can assume three different values. Hence, it was converted into three binary features: Asia ($Origin\ Asia$), Europe ($Origin\ EU$), and North America ($Origin\ NA$).

Table A.1: Selected instances for local explanation from the Gaussian data set. We picked the instances of lowest and highest target values.

	σ	θ	**Output**
Low	0.215	1.843	0.000
High	0.334	−0.026	1.189

Table A.2: Selected instances for local explanations from the Auto-MPG data set. We picked the instances of lowest and highest target values.

	Cyl.	*Disp.*	*HP*	*Weight*	*Acc.*	*Model year*	*Origin*			*MPG*
							Asia	*EU*	*NA*	
Low	8	400	150	4997	14	73	1	0	0	11
High	4	91	67	1850	13.8	80	0	0	1	44.6

To illustrate local explanations, we have selected two instances from each data set representing a *low* and a *high* value of the output feature. Tables A.1 and A.2 show the selected instances. For the Auto-MPG data set, we choose a car with a *low* output value of the target feature (representing a high fuel consumption car) and another with a *high* output value (representing an economical car in terms of fuel consumption).

A.2 Methods

We chose 10 methods to have their explanations evaluated, which include both white-box (2 methods), gray-box (one method), and post-hoc model agnostic methods (7 methods). These methods are indicated in Table 2.1 with an (*), together with its main characteristics. We have also added our agnostic method of explaining ELA in this comparative study. To select these methods, we take into account the widespread use in the xAI community, the availability of the implementation, and a detailed description of the applied methods. Finally, we have selected methods that use different explanation strategies to evaluate how different these explanations are and whether they provide different insights into the models' internal mechanisms[1].

Recall that post-hoc methods are used to explain a black-box model. We chose the Kernel Regression model as a representative of black-box models for regression. This model is reported to have a competitive accuracy compared to other black-box models and symbolic regression models [70]. The Kernel Regression algorithm learns a function

[1]The technical details about the installation and usage are presented in Appendix B.

(usually non-linear) in a kernel-induced space.

Since Kernel Regression models transform the original data into the Kernel space, we cannot easily interpret the generated model. This is when post-hoc explanation models come in.

A.3 Setup

This work focuses on explanations and how they can be interpreted. However, the error of the model needs to be taken into account before the explanation process starts. To do so, we chose not to perform the cross-validation procedure but only divide dataset into two parts: the training set and the test set. For the Gaussian dataset, we have generated $1,000$ training instances and 100 test instances. The data was generated without noise to avoid adding artifacts to the data. For the Auto-MPG dataset, we split the dataset, and we used 279 instances for training and 119 for test ($70 - 30\%$).

Thus, we use the training sets to build four models: a linear model and a decision tree (as representatives of intrinsically interpretable/white-box models), a symbolic regression model (representative of a gray-box model), and a kernel regression model (representative of a black-box model).

We have adjusted the linear model using Ordinary Least Squares for both data sets to minimize the residual sum of squares between the linear model and the training outputs. We have also added an intercept term to add more flexibility to the model. For the decision tree, we created the models limiting the tree to a depth of 4 since our main objective is interpretability.

For the Symbolic Regression (SR) method, we use the Genetic Programming with Non-linear Least Squares (GP-NLS) algorithm available with the Operon Framework[2]. This algorithm is an volutionary process to find the best equation representing the training instances. A binary tree represents each solution/individual. We set the hyper-parameters to generate simple and interpretable expressions, forcing the expressions to have a small size while still having a certain degree of freedom during the optimization process. We used only arithmetic operations and simple terminal nodes as the building blocks for the expressions ($[+, -, \times, /, constant, variable]$). The method was run for $1,000$ generations with 500 individuals in the population. The optimization process minimized $RMSE$ and the maximum tree depth and length of 4 and 15, respectively.

For the black-box model, we used the Kernel Regression provided by the Scikit-Learn library in *KernelRidge* package. For the parameters, we performed a grid search

[2]Available at https://github.com/heal-research/pyoperon.

Table A.3: RMSE obtained using the test set for both synthetic and real data sets. The best value for each data set is highlighted in bold.

Model	Gaussian	Auto-MPG
Linear Model	0.145	3.676
Decision Tree	0.071	3.897
Symbolic Regression	0.053	**3.209**
Kernel Regression	**0.012**	3.304

in each data set where the kernel ($'poly', 'polynomial', 'rbf', 'laplacian', 'sigmoid', 'cosine'$) and the values for alpha $(0.05, 0.1, 0.5, 1.0, 10.0)$, degree $(1, 2, 3, 5)$ and coef0 $(0.0, 0.05, 0.1, 0.5, 1.0, 10.0$ are varied. For Gaussian dataset the optimal parameters were: alpha=0.05, coef0=0, degree=1, and kernel=*laplacian*. For the Auto-MPG dataset, the best values found were: alpha=0.05, coef0=10.0, degree=2, and kernel=*poly*.

Thus, as a reference of the prediction error of the regression models, Table A.3 reports the root mean squared error (RMSE) obtained by each model on the test set. As we can see, intrinsically interpretable models are not always able to obtain a good result in relation to error. As the model becomes more complex, better results are obtained. However, to understand a complex model, we need auxiliary explanation methods.

A.4 Interpreting white-box models

Here we present explanations from a linear model and a decision tree model for each data set.

A.5 Gaussian dataset

The linear regression algorithm returned the following model, and the 95% confidence interval (CI) for each one of the coefficients is reported in Table A.4 :

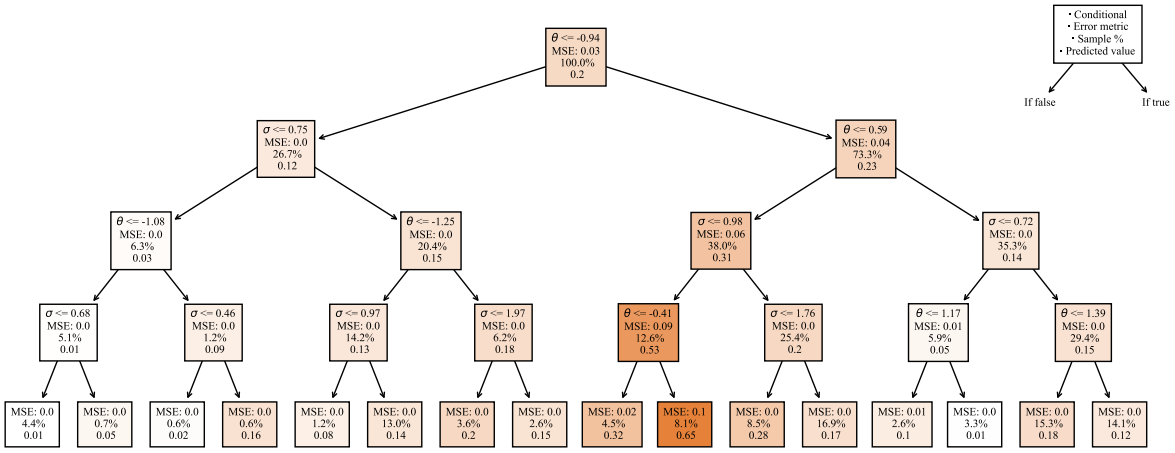
$$f_{\text{Linear}}(\theta, \sigma) = -0.0606\sigma - 0.0001\theta + 0.295. \tag{A.2}$$

This model can be read as: whenever $\theta = \sigma = 0$, we have a base value of 0.295, *i.e.*, the probability of observing $\theta = 0$ with zero mean and variance, is 0.295. Since,

Table A.4: Coefficient values and corresponding Confidence Intervals of the Linear Model obtained for the Gaussian data set.

	Coef	95% Confidence Interval
σ	-0.0606	$[-0.074, -0.048]$
θ	0.0001	$[-0.009, 0.009]$
Intercept	0.2954	$[0.272, 0.319]$

Figure A.1: Decision tree created from the Gaussian data set (Depth = 4). Colors indicate the extremity of the predicted value, with darker colors indicating higher values.



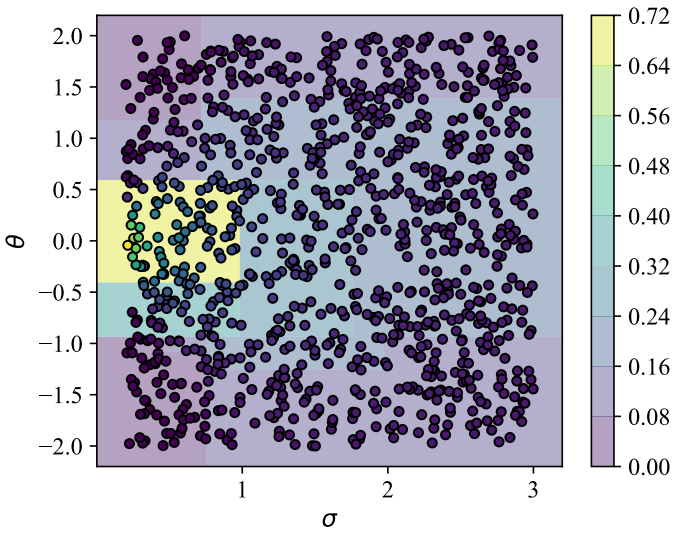
Source: created by the author.

in practice, this is an unrealistic situation (we cannot measure this probability for zero variance), we can fix a common value for σ to understand the intercept. For the common setting of a normal distribution $\mathcal{N}(0, 1)$, we have that the probability of observing $\theta = 0$ is 0.2344. Starting from the baseline $\mathcal{N}(0, 1)$, we can say that every unitary increase in σ will decrease the probability of observing a value of θ by 0.0001. On the other hand, if we change the value of θ by one unit, we will see a reduction of 0.06 in the probability, which is true when we are moving away from 0. However, if we decrease the value of θ, the linear model will increase the probability.

The CI represents the interval having 95% probability of containing the actual value of the coefficient of the linear model (since the model is derived from the sampled data, thus have an uncertainty associated with the samples). From the CI of each feature, θ contains zero as part of the interval. This could be interpreted as having a high probability that θ could be discarded from the model by assigning zero to its coefficient.

Looking at a different interpretable model, Fig. A.1 shows the tree created. By inspecting the rules generated, we observe some trends such as, the closer to zero the θ value is, the larger the prediction. This makes sense since our data set includes only positive values of σ and, the closer to $\theta = 0$, the higher the probability. On the other hand, a higher value for the standard deviation reduces the probability for any given θ.

Figure A.2: Gaussian data set: decision surface (Tree depth = 4). The color of the sub-regions indicates the prediction value make for the samples within the sub-regions.



Source: created by the author.

This can be better visualized by plotting the decision surface, as in Fig. A.2. As we can see, the highest values (yellow color) are closer to where θ and σ are the smallest. Also, the rate of change induced by θ is faster than that for σ.

The non-linear model generated by the decision tree is more accessible to a lay audience and presents a smaller error than the linear regression. These interpretable methods provided, with no additional tools, the possibility of understanding the relationships between the input features, allowing the user to obtain insights about the problem. With this information, it is also possible to know how to change the inputs to produce the desired output.

Despite being considered white-box models, they do not scale well for larger problems, as the learned structures tend to be more complex. Also, the non-linear relationships of larger problems may not be approximated accurately, resulting in a model that fails to adequately describe how y responds to $\mathbf{x}$. Tree pruning and different types of regularization can help mitigate this problem but are out of the scope of this work.

A.6 Auto-MPG dataset

For the Auto-MPG dataset, Table A.5 reports the coefficients and confidences intervals. The obtained linear model is shown in Eq. A.3.

Table A.5: Coefficient Values and corresponding Confidence Intervals of the Linear Model obtained for the Auto-MPG data set.

	Coef	95% Confidence Interval
Cyl.	-0.333	$[-1.062, 0.395]$
Disp.	0.02	$[0.002, 0.038]$
HP	-0.005	$[-0.03, 0.019]$
Weight	-0.007	$[-0.009, -0.006]$
Acc.	0.115	$[-0.9, 0.32]$
Model year	0.762	$[0.649, 0.876]$
Origin Asia	-1.259	$[-7.774, -2.567]$
Origin EU	0.457	$[-5.877, -1.034]$
Origin NA	0.802	$[-0.598, -0.521]$
Intercept	-15.648	$[-19.022, -4.449]$

$$\begin{aligned} f_{\text{Linear}}(\mathbf{x}) = &- 0.333 \cdot Cyl. + 0.02 \cdot Disp. - 0.005 \cdot HP \\ &- 0.007 \cdot Weight + 0.115 \cdot Acc. + 0.762 \cdot Model\ year \\ &- 1.259 \cdot Origin\ Asia + 0.457 \cdot Origin\ EU \\ &+ 0.802 \cdot Origin\ NA - 15.648. \end{aligned} \tag{A.3}$$

Observing the intervals of the coefficients, we can see that the current samples are insufficient to determine if the variables *Cyl.*, *HP*, and *Acc.* have a positive or negative effect (or even any effect) on the output. This should be taken into consideration when explaining the model.

For this data set, we do not have the ground-truth. The base case here, where the value of each feature is equal to 0, does not bring any information once this scenario does not happen in the real-world. Not surprisingly, this situation would render a negative fuel consumption. We can interpret the intercept using the average values of each feature; when evaluating this model using the average of each feature, we have a prediction of fuel consumption of 22.91 as the baseline.

The model in Eq. A.3 also tells us that whenever we increase the cylinder (*Cyl.*) by one, we are likely to observe a decrease of 0.33 in fuel consumption (output). We also observe a reduction in fuel consumption when we increase horsepower (*HP*) and weight (*Weight*) and leave all other features constant. But note that we should be careful with interpreting a linear model as it does not automatically capture the interaction. Another conclusion is that whenever we increase the horsepower, we also increase the acceleration (*Acc.*). As such, we should further investigate the impact of a joint increase in both features to draw a definite conclusion.

Another point we should be aware of when analyzing a linear model is that when the features are not on the same scale, it can be counter-intuitive to measure the importance of each feature by their linear coefficients.

Turning to the decision tree model, shown in Fig. A.3, note that here we can see how the features interact to make a prediction. To illustrate, taking the leftmost path from the root to the leaves leads to a prediction where we have a small value for displacement, horsepower, model year, and weight, leading to an average consumption of 29.91. On the other hand, newer models (model year) will increase the output (decrease fuel consumption). This trend appears in every branch and suggests further investigation on what has changed throughout the years to reduce consumption. These results suggest a new experiment with a different regression model built for every decade. The lowest fuel consumption is obtained with high values for displacement, horsepower, and weight; and a low value for the model year.

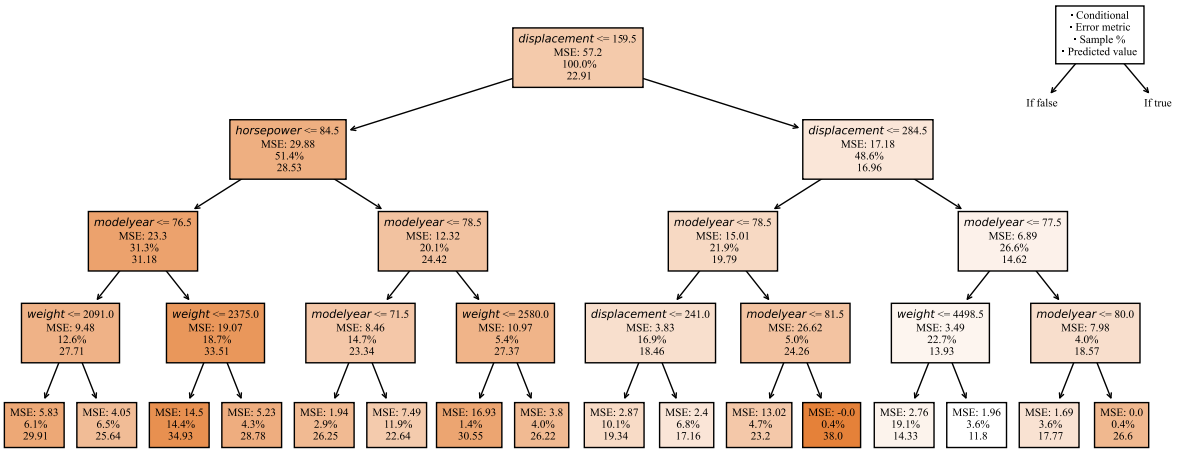
A.7 Linear model *vs* Tree-based model

Comparing both white-box models, the linear model returns a more straightforward interpretation where it is quite clear the expected (or a rough approximation of) association between each feature and the target. Using this model, we can make decisions on how to change the input to obtain the desired change in the output of the system. Besides that, this type of model has many well-established statistical tools to give more confidence in user decisions. In particular, we can easily calculate an interval for the parameters and support decisions with the average and extreme cases using both ends of the boundaries. These boundaries also show whether there is a chance that the effect of that feature could be only caused by random chance.

One downside of this model is that it does not account for non-linearities and interactions of the features. The regression tree model can capture such properties of the model and still provide a clear interpretation of the decision process. But, unlike linear models, the focus on interpretability is only on how the decision is made and does not add any information about the association of a particular feature to the output.

If the main goal is to understand the studied system, both approaches could give different insights. While a linear model provides a better understanding of the expected behavior of the system, the regression tree can reveal interactions and non-linearities that could be incorporated into the linear model.

Figure A.3: Decision tree created from the Auto-MPG data set.



Source: created by the author.

A.8 Interpreting gray-box models

While the models discussed in the previous Section were inherently interpretable, gray-box models, as the name suggests, are within black and white-box models. To illustrate the interpretation of these models, we chose the Symbolic Regression (SR) method. Here we present the explanations obtained for each data set.

A.9 Gaussian dataset

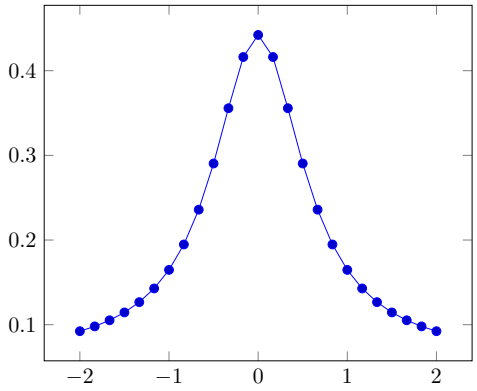
SR has the advantage of finding non-linear expressions that can still be inspected to obtain insights from the problem domain. For the Gaussian data set, the returned expression was:

$$f_{\text{SR}}(\sigma, \theta) = \frac{0.146}{0.381\sigma + \theta^2} + 0.059. \tag{A.4}$$

Since this is a non-linear model, we cannot make the same interpretation of the coefficients as we did with the linear one. Yet, we can still fix the values of some of the features to analyze the behavior of the function.

A first baseline is to set $\theta = 0$ and $\sigma = 1$, which returns 0.442 (the real value should be 0.399). We could also calculate the baseline with the average value of each

Figure A.4: Symbolic regression model (output presented in y axis as a function of θ) described in Eq. A.4 fixing $\sigma = 1$.



Source: created by the author.

feature to see the prediction for the average case. By fixing $\sigma = 1$ we can plot the behavior for different values of θ, as illustrated in Fig. A.4. This visualization can help the practitioner to inspect and debug the model to analyze whether it is misbehaving at certain points or to understand the dynamics of the function.

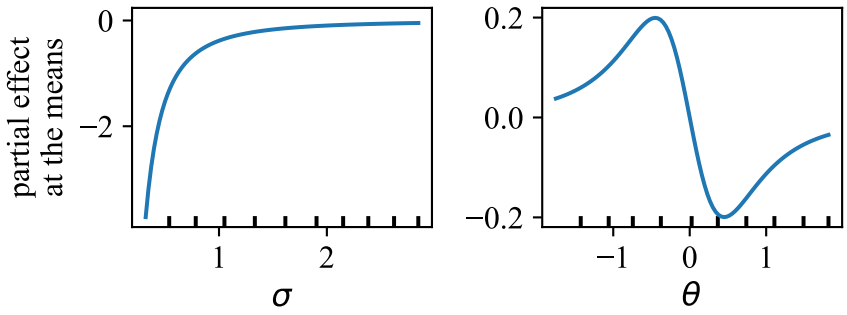
When we calculate the partial derivative w.r.t. θ and by fixing σ at the average value (known as the Partial Effects at the Means [10]), we can gain a deeper understanding of the model behavior, as seen in Fig. A.5.

We observe from the left plot that, when fixing the value of θ at the mean, varying σ from 0 to 1 will have a negative effect on the probability of observing this θ value. This is intuitive with the expected properties of the Gaussian function: as we increase the value of σ we observe a flatter curve. We can also see that after some point, the smaller the *rate of flattening* of the curve, the higher the variance (approximately by a logarithm factor). The plot to the right shows the partial effect when fixing σ while varying the value of θ. At $\theta = 0.0$, we have a local optima point (from Fig. A.4 it is a point of maxima) and, for every positive value of θ, whenever we increase its value, we observe a decrease in the target value (the partial derivative values are negative).

On the other hand, if we have a negative value of θ and increase its value, we move the target value closer to the optima. Also, observe that the intensity of increase/decrease is maximum at approximately $\theta = \pm 0.3$, and, as θ moves away from 0, the prediction gets closer to 0, indicating a reduced effect on the target. Hence, the symbolic model gives another perspective of the model dynamics, with better insights into how the studied system behaves.

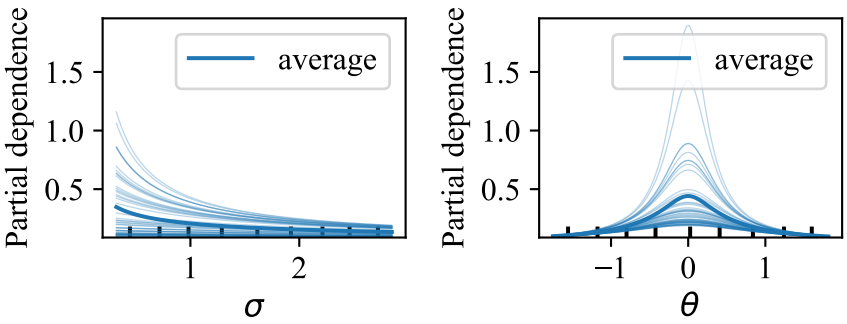
In Fig. A.6 we also show the Partial Dependence plots for the Gaussian data set. A partial dependence plot is a graph in which the x-axis represents the values for one of the features, and the y-axis represents the expected output for that particular value when the remaining features are kept constant at a baseline value (usually the average). We can say that these plots give the same information as the partial effect but from a

Figure A.5: Partial Effects at the Means for the Symbolic Regression in the Gaussian data set.



Source: created by the author.

Figure A.6: Partial Dependence Plot for the Symbolic Regression in the Gaussian data set.



Source: created by the author.

different angle.

Note, from the left plot, that for all of the observed values of θ, the probability decreases as σ increases (due to the flattening of the curve). Likewise, we can see the Gaussian curve for different values of σ on the right plot.

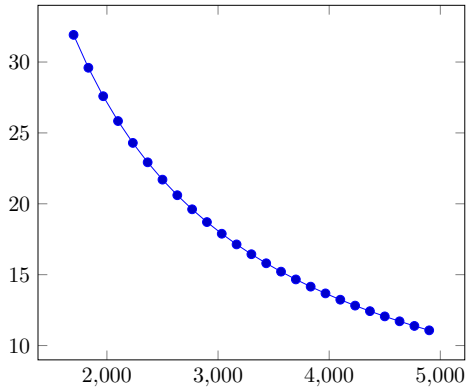
A.10 Auto-MPG dataset

For the Auto-MPG data set, the symbolic regression model obtained was:

$$f_{\text{SR}}(\mathbf{x}) = 0.106 \cdot \frac{(Model\ year)^2 \cdot (Acc. + Model\ year + Origin\ EU)}{Weight} + 2.275. \tag{A.5}$$

The SR model could perform an implicit feature selection by using only a subset of the problem features. Four of them are part of the model: *Weight*, *Acc.* (acceleration),

Figure A.7: Symbolic regression model (output presented in y axis as a function of $Model\ year$) described in Eq. A.5 fixing all features except $Model\ year$.



Source: created by the author.

$Model\ year$, and $Origin\ EU$ (country of origin: Europe). By analyzing the Eq. A.5, we can see an interaction between multiple features.

Again, we can see the importance of the model year having a quadratic strength. It interacts with the linear combination of the values of acceleration ($Acc.$), year of the model ($Model\ year$), and European cars ($Origin\ EU$), meaning that these features contribute to increasing the predicted MPG. The feature weight ($Weight$) acts by decreasing the value of MPG prediction.

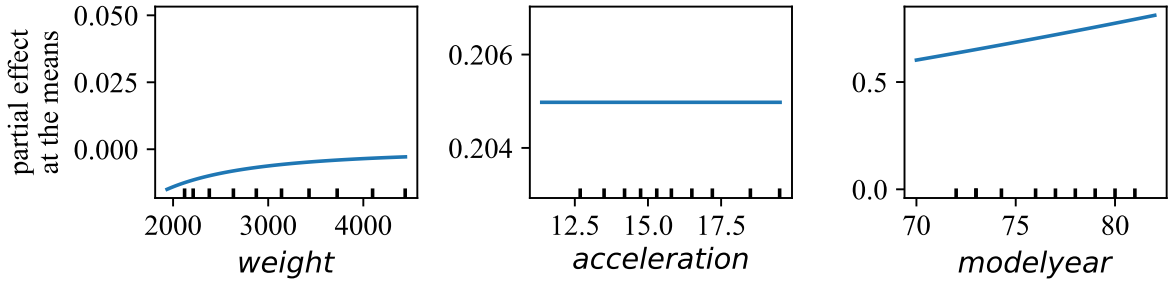
It is plausible to assume that as the model year increases, it exerts a quadratic effect over the combination of the acceleration of the car, having an additional impact if the country of origin is in Europe. One can interpret this as if the feature model year captures the improvements made by engineers over the years to increase the car's performance. A car with a higher weight decreases MPG, and an increment on this feature would likely result in a less efficient vehicle.

We can apply the same treatments as before, first evaluating the model at the average values of each feature, resulting in a baseline value of $16,277$. We can also fix the values of all features but one and inspect how the remainder feature behaves. To illustrate, we can fix $Acc. = 15, Model\ year = 75, Origin\ EU = 1$ and observe the behavior of the weight, as illustrated in Fig. A.7. This figure shows that when keeping everything else constant, the *mpg* reduces logarithmically as the car weight increases, indicating a consumption increase.

It is essential to highlight that we should be careful before making conclusions about these models. We must understand the limitations and use these analyses as a *stepping stone* for further investigation. Regarding the model year, we should be aware that the data was collected during a small time-frame (between 1970 and 1982). Hence, we should be careful when extrapolating it to recent years as we have had many advancements in engineering.

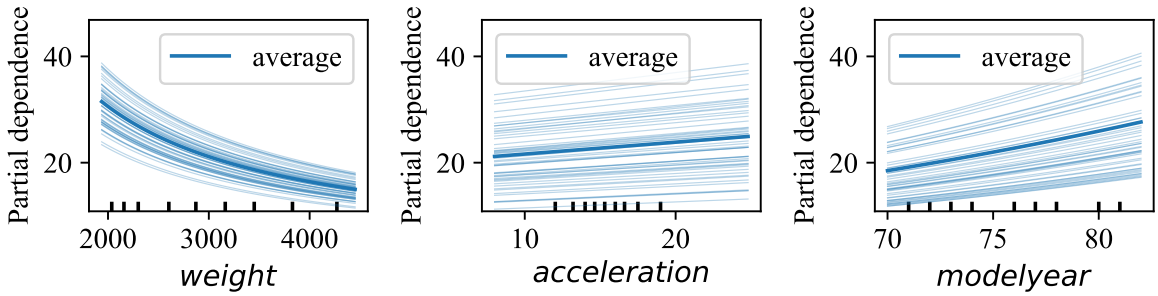
The Partial Effects at the Means for three features of the Auto-MPG data set are

Figure A.8: Symbolic Regression global explanations using the Partial Effects for the Auto-MPG data set.



Source: created by the author.

Figure A.9: Symbolic Regression global explanations using the Partial Dependence Plots (average) for the Auto-MPG data set. The Individual Conditional Expectations (ICE) are represented in lighter lines.



Source: created by the author.

illustrated in Fig. A.8. As we can see from these plots, the effect of weight is always negative, and it decreases at a quadratic rate. The impact of the acceleration, on the other hand, remains constant, as it has only a linear influence on the prediction. For the model year, the effect is positive and increases at a linear rate.

In Fig. A.9 we can see the PDP as a counterpart of the Partial Effects, showing that, for the weight feature, there is a steeper decrease in mpg when increasing the weight from 2000 to 3000 than when increasing from 4000 to 5000, as expected from the partial effect analysis.

As we saw, SR has the advantage over traditional regression techniques that it can find interactions of the features that best minimize the prediction error. The structure is modeled and adapted during the search. The final equations can be interpretable depending on several factors: domain knowledge, the function form, and the final expression size. The advantage of having a mathematical expression is that we can use the concept of Partial Effects to obtain feature importance explanations based on the actual model and not on simply changing the input and observing the effects on the output — making the Partial Effects an explanation that corresponds to the decision process of the model.

A.11 Gray-box *vs* White-box models

The Symbolic Regression model extends the linear model with the possibility of describing the non-linearities of the studied system. Also, it is capable of automatically performing feature selection and the identification of interactions among features. As such, the SR model can deliver the same insights described in the white-box section. However, given that the model can be non-linear, the interpretation of the numerical parameters, analogue to the linear model, may not be straightforward. With a linear model, the association of a given feature to the target is constant, on a non-linear model, it can depend on the values of the other features. In this case, it is necessary to summarize this information with aggregation measures and extract the insights with the help of visual aids (e.g., PDP and Partial Effect plots).

Hence, instead of having an generic explanation for every point, we have to interpret every data point individually or in a limited region at a time. Even though the Partial Effects at the mean and the PDP plots carry the same information, they do so from a different point of view. While PDP shows the expected value of the target for a given value of a feature, the Partial Effects plot shows the impact of an infinitesimal variation of that variable, closer to the interpretation of a linear model.

We argue that both views of the model are important for the understanding and decision-making process. While we can identify regions that contain values of interest for the target (with the PDP plot), we can also find the critical regions that vary the model response the most (with the Partial Effect plot).

A.12 Post-hoc explanations of black-box models

As previously explained, post-hoc methods are used to explain a black-box model. We chose the Kernel Regression model to represent a black-box model, as it learns a function (usually non-linear) in a kernel-induced space using the given data set as input. Since Kernel Regression models transform the original data into the Kernel space, we cannot easily interpret the generated model as we have done so far. For that, we need the support of explanation methods.

We have selected 9 agnostic post-hoc explanation methods to be evaluated, including methods following all the main approaches previously proposed in the literature. We understand that no method of explanation can provide a complete understanding of the

model. At the same time, we believe there is no need for competition between those who explain more or best — we believe that the concomitant use of different methods can be beneficial so that the user can trust, improve or have new insights into the scenario under study.

We can say that the SHAP, LIME, SAGE, Partial Effects, and Integrated Gradients method can return the importance of each feature to obtain the output of the regressor. SHAP and Partial Effects produce global and local importance. For instance, LIME and Integrated Gradients only the local importance and SAGE only the global importance. Additionally, SHAP, LIME, and Integrated Gradients also report on the impact that can be caused on the output by changing the observed input values. The ELA method can show us the importance of local features and also, through a visual explanation, the importance followed by various output intervals of the regressor. In addition, we were able to obtain from ELA information about feature values that can be changed without significantly changing the output.

From a visual explanation, the ICE/PDP methods can show the change in the values in each feature caused in the model output. ProtoDash provides its explanation by selecting the most relevant and diverse instances for the task, with information about the weight of each one. Finally, DiCE explains which features can be minimally changed so that our output is substantially modified elsewhere in the solution space.

This section compares the explanations provided by the different methods of explanation, highlighting what we can learn from each of them and checking for possible agreements or contradictions. For each data set, we organize our discussions according to the scope of the explanation, i.e., global versus local.

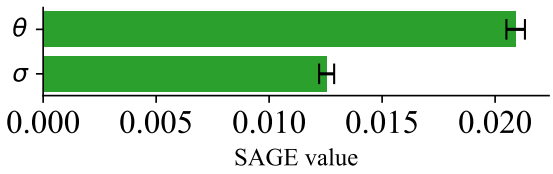
A.13 Explanations for the Gaussian data set

A.13.1 Global explanation

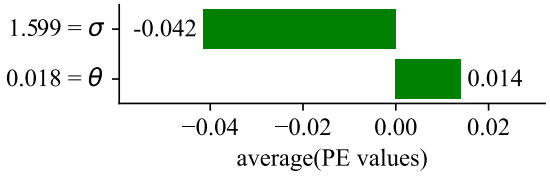
Global explanation methods attempt to provide a single explanation to summarize the global behavior of the model. Although there are only two features for the Gaussian data set, we start with feature importance methods and discuss why this dataset is challenging for them.

Having only two features, three scenarios are possible for global feature importance explanations: both are equally important, or each one of the two features can have a more

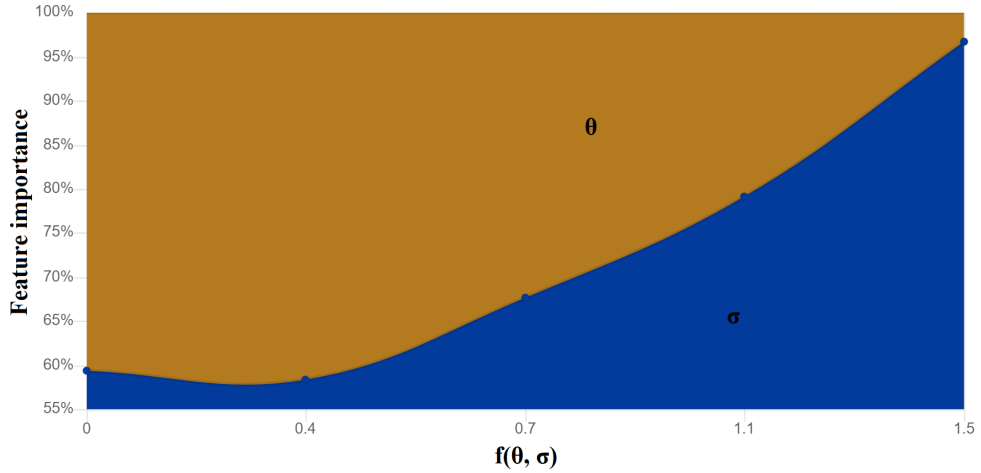
Figure A.10: Explaining the Kernel Regression model globally for the Gaussian data set.



(a) SAGE feature importance.



(b) Partial Effects feature importance.



(c) ELA feature importance by prediction output value.

Source: created by the author.

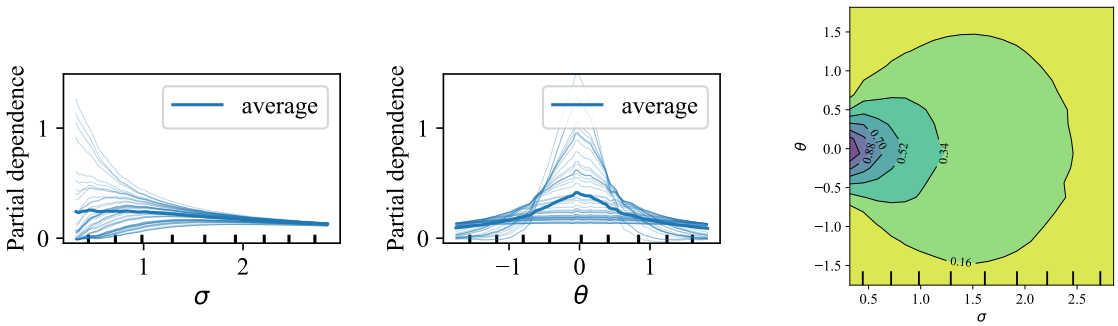
significant impact than the other. The ground-truth equation calculates the probability of observing θ for a Gaussian distribution centered on zero with a standard deviation σ.

We notice that exists a relationship of dependence between θ and σ: the closer θ is to the extremes of the distribution (*i.e.*, $\pm 3\sigma$), the less its values impact the predicted probability, since $\approx 99\%$ of the data in a Gaussian distributions lies within $[-3\sigma, 3\sigma]$. This means that as θ gets closer to $\pm\sigma$, the output should suffer less from variations in θ. Notice that the feature values for θ are in the range $[-2, 2]$ and for σ they are in $[0.2, 3.0]$. Hence, when θ assumes a value closer to the distribution center ($\mu = 0$), σ becomes less important. σ impact, on the other hand, decreases as its values get larger than θ. This interaction of features makes this data set challenging to global explanation methods, which we expect to retrieve this dynamic or at least have an acceptable approximation.

Fig. A.10 shows the global explanations obtained by SAGE (Fig. A.10a), Partial Effects (Fig. A.10b), and ELA (Fig. A.10c). We first notice that SAGE and Partial Effects — which support global feature importance explanations — show different feature importance ranks. SAGE considered θ to have nearly twice the importance on the prediction than σ, while Partial Effects showed the opposite – σ is more important than θ.

Although ELA provides a visual explanation, this explanation is based on feature importance, and hence we compare it with the previous feature-based approaches. Ob-

Figure A.11: Kernel Regression global explanations using the Partial Dependence Plots for the Gaussian data set.



(a) Single features. (b) Features interaction.

Source: created by the author.

serve that the interaction between terms is better represented by ELA, but it is given in terms of the output. As the output increases (x axis), the contribution of the different features changes — this leads to a conclusion that the model behavior is non-linear and thus cannot be simply generalized by global feature importance methods.

Finally, Fig. A.11a shows the partial dependence plots (PDP) for each feature, with the lighter lines representing the Individual conditional expectations (ICE), while Fig. A.11b considers both features interaction. The PDP individual plots show that, on average, σ has slightly decreasing importance as it grows. In contrast, θ average behavior acts as a radial basis function: the importance depends only on the distance to the origin. However, the ICE lines indicate that exists many samples in the data set with high deviation from the average behavior. It is known that the average is a biased estimator that can be influenced by outliers and data distribution. Nevertheless, PDP plots complement the ELA plots by elucidating the feature importance segmented by their interval, not by the predicted value. This is because ELA generalizes multiple plots into one.

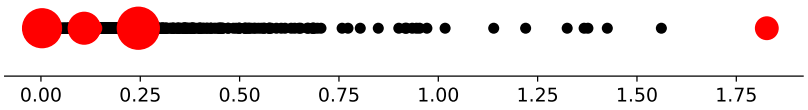
We can visualize the interactions between the two features when plotting the PDP shown in Fig. A.11b. The feature θ still has a radial basis behavior, but as σ increases, the spread of the radial function also increases. While PDP provides a helpful explanation, it assumes feature independence. The PDP for a single feature is calculated by fixing the other features in representative values, eliminating existing correlations. Also, the maximum number of features we can plot simultaneously is two.

Turning to prototype-based methods, for ProtoDash, we generated 7 prototypes to display in Table A.6 In addition to the features of each prototype, we also obtain their weights. The weight quantifies how well each prototype can represent the data. The prototype, with a weight of nearly 50%, indicates that $\sigma = 1.62$ and $\theta = -0.05$ strongly represent the overall data set. It corroborates with the fact that $\theta = 0$ has the minimum

Table A.6: Top seven prototypes selected to represent the global behavior of the Kernel Regression for the Gaussian data set. The weight leverages the importance of each prototype to summarize the whole data set.

# **Prototype**	1	2	3	4	5	6	7
σ	1.626	0.604	0.456	2.990	2.919	0.213	2.996
θ	−0.057	1.998	−1.996	1.909	−1.761	−0.043	0.262
Output	0.245	0.002	6.159e-05	0.108	0.113	1.826	0.132
Weights (%)	0.464	0.124	0.107	0.107	0.104	0.072	0.041

Figure A.12: Target feature space highlighting the instances (in red) chosen as prototypes by Protodash for the Gaussian data set. The black dots represents the other samples from the data set.



Source: created by the author.

average distance between all data points because of the function symmetry. The following four prototypes have weights around 0.1, each representing an extreme point of the data set, where either θ or σ is close to the extreme values of their respective domains.

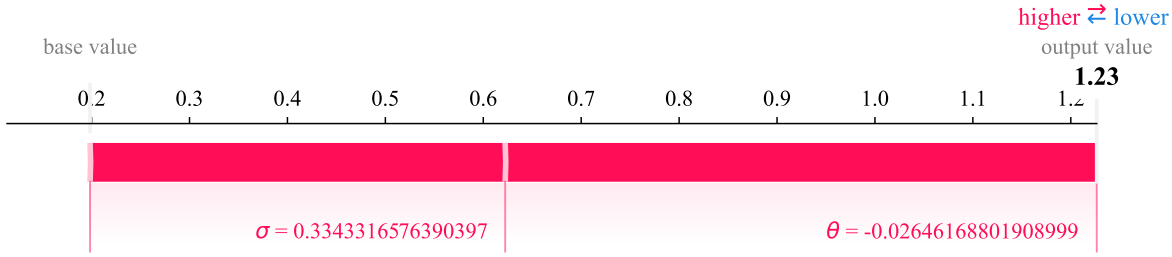
Fig. A.12 shows the prediction for each observation in the training partition, with red dots indicating the prototypes. The size of the dot means the weight of the prototype. We can see that most prototypes have similar output values but very different feature values.

A.13.2 Local explanation

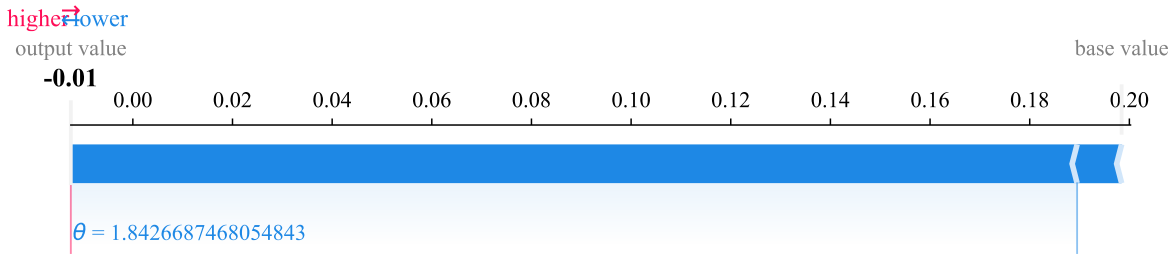
In contrast with global explanation methods, local explanation methods provide different explanations for every single instance (observation) of the data set. In this section, we chose two instances to analyze the explanations provided by different methods: one has a small and the other a large prediction value.

We start with methods based on feature importance. Since describing a single output is a more straightforward task than summarizing a (possibly complex) prediction model, feature importance explanations are expected to play a more significant role in the local context than in the global. Fig. A.13 shows the local explanation provided by SHAP for the selected examples.

Figure A.13: Kernel Regression local explanations using the SHAP explainer for the Gaussian data set.



(a) Local explanation for the lowest target value



(b) Local explanation for the highest target value

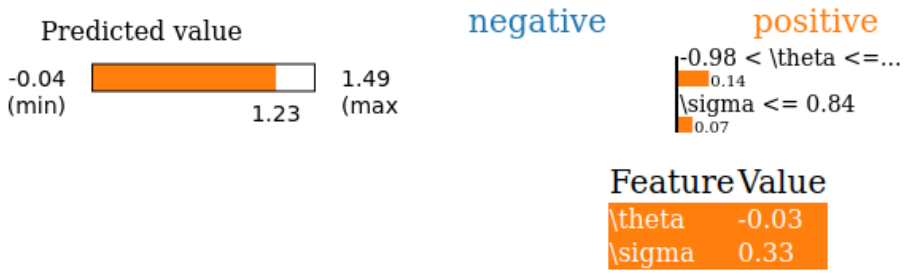
Source: created by the author.

The SHAP plots are called *force plots* and visually show the contribution of each feature to that particular prediction. In the plot, the blue/red highlights represent a decrease/increase in the output when compared to a baseline prediction, which considers all features as having their average observed values. We can also observe the ability of SHAP to capture the non-linearity of the data when comparing the numerical difference of contributions to two local explanations. Through the plots, SHAP shows that, for the high probability observation, to go from the baseline (0.26) to the predicted output (1.23), both σ and θ have a positive contribution, with θ being slightly more significant than σ.

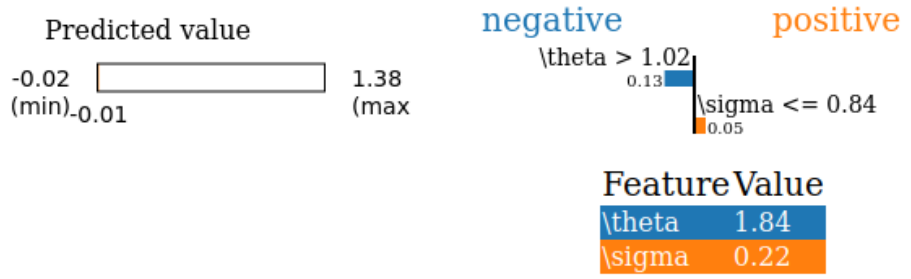
For the low prediction observation, in contrast, SHAP shows σ contributed positively with a small strength, and θ contributed positively with a Shapley value of 1.842. Hence, σ has more significant importance for the high probability prediction than for the low probability for the compared samples.

Next, Fig. A.14 reports the local explanations given by LIME. In this case, the blue bars represent the negative contributions, and the orange ones represent the contributions to increase the output of the specific test instance feature. Note that the positive or negative contribution information returned by the SHAP and LIME methods cannot be directly compared as the reference point is not the same. Thus, LIME tells us that, since $\sigma \leq 0.84$, its local importance is 0.036. Similarly, since $\theta > 1.02$, its local importance is -0.124. This reflects the local association of these specific values. Hence, at this point,

Figure A.14: Kernel Regression local explanations using the LIME explainer for the Gaussian data set.



(a) Local explanation for the lowest target value



(b) Local explanation for the highest target value

Source: created by the author.

Table A.7: Kernel Regression local explanations using the ELA explainer for the Gaussian data set. ELA reports the normalized feature importance by approximating the black-box locally using an interpretable linear model.

Low output	**High output**
y': -0.010	y': 1.213
Coefficients: $[0.063, 0.013]$	Coefficients: $[-2.376, -0.010]$
intercept: -0.049	intercept: 2.008
Importance of features:	Importance of features:
σ : 82.075	σ : 99.531
θ : 17.925	θ : 0.469
Local variation: $-0.008 - 0.003$	Local variation: $1.186 - 1.186$
$0.215 \leq \sigma \leq 0.453$	$0.334 \leq \sigma \leq 0.346$
$1.600 \leq \theta \leq 1.953$	$-0.05 \leq \theta \leq -0.02$

increasing θ will have a negative effect on the output.

Similarly, Table A.7 shows the local explanations provided by ELA. For both instances, ELA reports σ as the most important feature, contrary to the previous local feature importance methods. Since this explainer is based on a local linear model, the interpretation is similar to a linear regression within the specified neighborhood. As we can see from this table, for the low output instance, both features have a positive effect, so any increase in the values of the feature will reflect an increase in the output. For the high output instance, both features have a negative effect.

Table A.8: Kernel Regression local explanations using the DiCE counterfactual (CF) explanations for the Gaussian data set.

	σ	θ	f
Original input high	0.334	−0.02	1.226
CF 1	0.278	0.566	0.205
CF 2	0.231	0.564	0.181
Original input low	0.215	1.842	−0.012
CF 1	0.231	0.564	0.181
CF 2	0.278	0.566	0.205

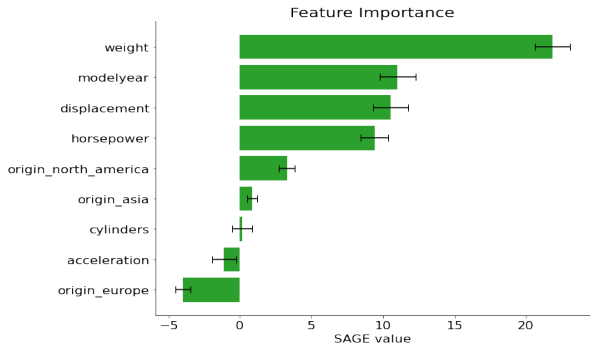
Finally, Table A.8 shows the counterfactual explanations created by DiCE. This explanation tries to find the smallest change in the original input that would change the prediction. We have set the goal of finding an input example that changed the output to 0.2. As we can see from the table, in both local instances, the main difference is to decrease/increase θ to a value of 0.56. This reveals the symmetry of the function where, starting from both extremes, they both should reach the same central value to get the same output of interest.

From the diverging explanations, we can conclude that SHAP, LIME, and ELA provide feature importance explanations that cannot be directly compared since they are calculated differently and according to a distinct referential: SHAP explains the prediction by showing the contribution each feature gives to go from the basal prediction to the actual output; LIME fits a local linear model over synthetic data generated around the observation, and uses its coefficients to explain the local behavior; and ELA uses k neighbors to fit the same linear model — instead of synthetic data sampled and weighted based on the observation neighborhood —, thus normalizing the coefficients to provide a more intuitive value to understand, since the explanations represent proportions.

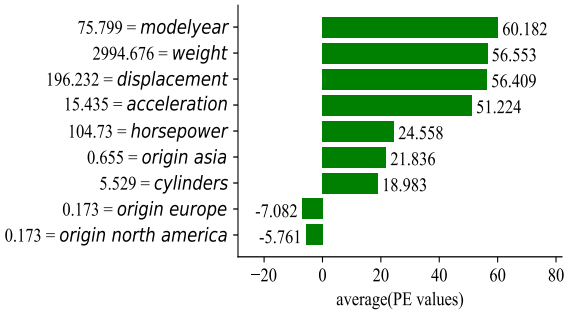
Which explanation should we use? This choice must be made taking into account what the user wants to know about the model. Is it "how to create minimal/maximal changes to the prediction?" In this case, LIME or ELA are better suitable; or is it "to understand which features, in contrast to the average observation, were most influential to the prediction?" In this case, SHAP provides a more meaningful explanation.

Neither the explanations are wrong, nor they provide contradictory results, but they show different perspectives of the same black-box model. The discussion about causality between explanations and the model's behavior is out of the scope of this work and is still an open question.

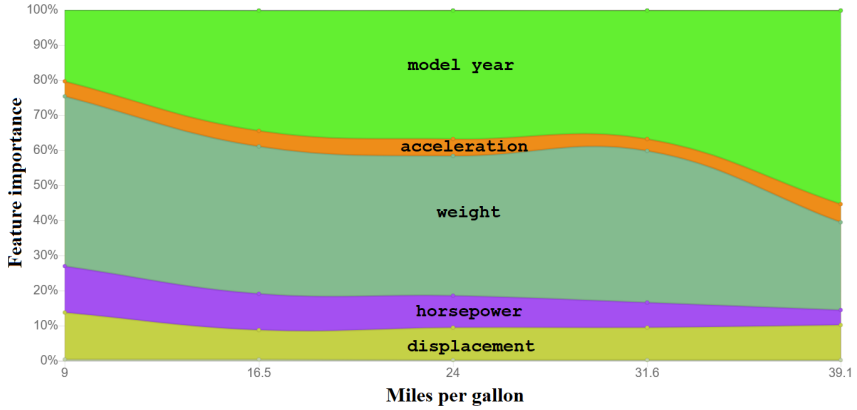
Figure A.15: Explaining the Kernel Regression model globally for the Auto-MPG data set.



(a) SAGE feature importance.



(b) Partial Effects feature importance.



(c) ELA feature importance by prediction output value.

Source: created by the author.

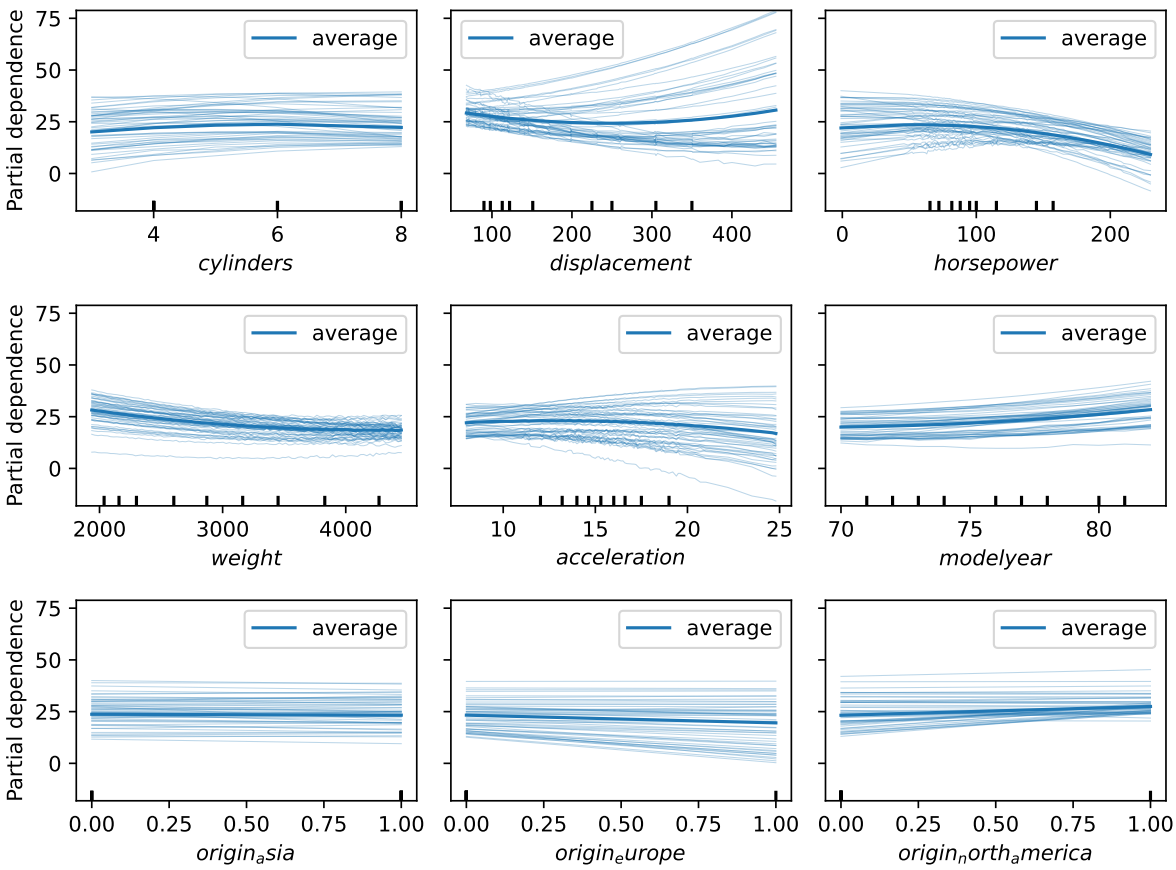
A.14 Explanation for the Auto-MPG data set

A.14.1 Global explanation

The results of the global agnostic explanations of SAGE, Partial Effects and ELA for the Auto-MPG data set applied to the KR model are reported in Fig.A.16. We must remember that these plots depict an aggregated measure of the local explanations from the sampled data available. As such, these plots can give a biased view of this importance as they will not reflect the reality for every data point. ELA plot alleviates this problem since it aggregates the importance per target value, as we will explain next.

Regarding the average relevance of each feature, all methods agree on the two most important features: weight and model year. The weight directly influences the inertia of the car, which makes it more challenging to get the car out of rest. From the explanations,

Figure A.16: Kernel Regression global explanations using the Partial Dependence Plots for the Auto-MPG data set.



Source: created by the author.

we conclude that a greater fuel consumption occurs due to the displacement as we have heavier cars, thus reducing the model's MPG output. The car model year shows that there has been an evolution in the manufacturing of vehicles, making them more efficient over time.

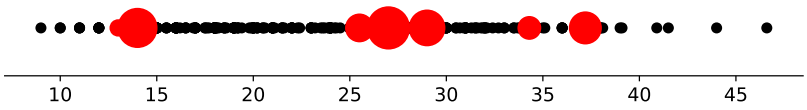
As previously noted, these plots describe an aggregated measure of importance, but they do not show the impact of each feature depending on their value. In non-linear models, the impact of a feature is not the same throughout the predictions, unlike in linear models. To tackle this problem, different from the bar plots generated by most explainers, ELA (Fig. A.15c) includes additional visual information about the aggregate importance w.r.t. the target value. In this particular plot, the weight is a much more critical factor for low MPG predictions. On the other hand, the high MPG predictions have more influence from the model year.

Concerning the information provided by the PDP plots (Fig. A.16), they display an average of the target value at each corresponding feature value, providing a more detailed view of the impact each feature has on the prediction. From this plot, we can see that a low cylinder has a negative impact on MPG. The features with the highest

Table A.9: Top seven prototypes selected to represent the global behavior of the Kernel Regression for the Auto-MPG data set.

# **Prototype**	1	2	3	4	5	6	7
cylinder	4	8	4	4	4	4	8
displacement	97.0	304.0	90.0	86.0	122.0	97.0	350.0
horsepower	88.0	150.0	70.0	65.0	96.0	78.0	145.0
weight	2130.0	3672.0	1937.0	2019.0	2300.0	2188.0	4055.0
acceleration	14.5	11.5	14.2	16.4	15.5	15.8	12.0
modelyear	71	73	76	80	77	80	76
Origin Asia	0	1	0	0	1	0	1
Origin EU	0	0	1	0	0	1	0
Origin NA	1	0	0	0	0	0	0
Output	27.0	14.0	29.0	37.2	25.5	34.3	13.0
Weights(%)	0.18	0.18	0.18	0.12	0.12	0.11	0.10

Figure A.17: Target feature space highlighting the instances chosen as prototypes for the Auto-MPG data set.

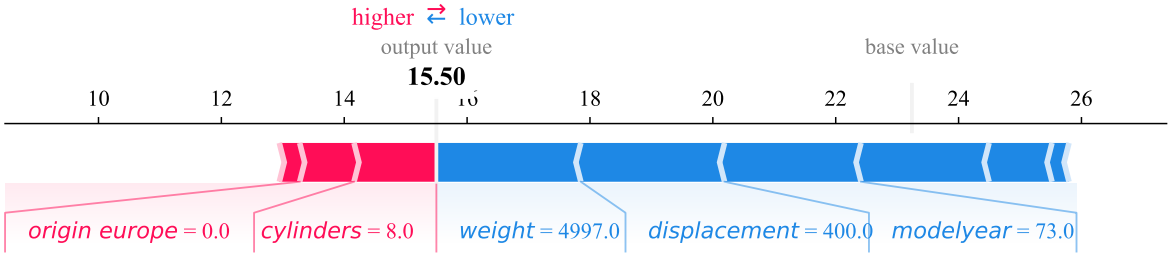


Source: created by the author.

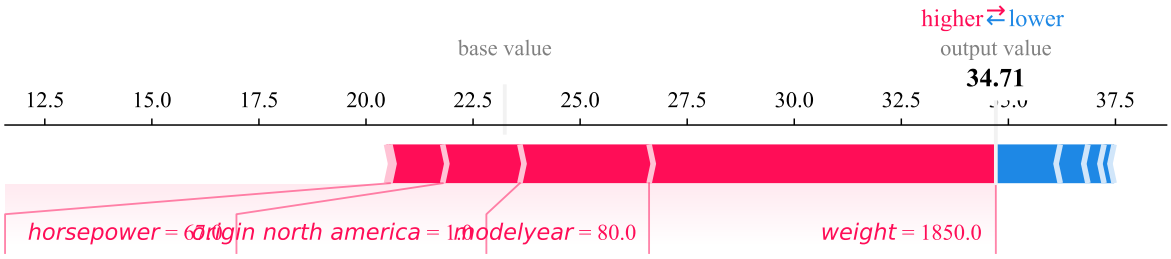
impact are the weight and model year, corroborating the previous explanations. Unlike the earlier plots, we can see that the impact of the weight is at its maximum at the lowest observed values. After 3000 Kg, we can observe a negative impact on the target feature. We can also plot the impact on a sample of different data points (ICE) instead of just the average effect, revealing whether the behavior of the model is consistent throughout the data instances or if there is any additional factor that should be considered. These plots show that horsepower and displacement present the most variation in behavior for different instances.

Again, we finish by showing the results of the prototype-based explanation generated by ProtoDash. ProtoDash selects prototypes in order, such that the next prototype guarantees new data particularities are discovered. The first seven selected prototypes are shown in Table A.9, which offers a significant variability in the selected instances. Furthermore, we can see that ProtoDash can explore the solution space well when viewing the graph in Fig. A.17, where it shows the outputs for the training instances in black and the outputs of the prototype points in red, with the largest circumferences being the most relevant prototypes. Thus, with ProtoDash, we learned about the most representative and diverse instances that contributed the most to the construction of the model, so we can understand the output obtained by new samples by analyzing its most similar

Figure A.18: Kernel Regression local explanations using the SHAP explainer for the Auto-MPG data set.



(a) Local explanation for the lowest target value



(b) Local explanation for the highest target value

Source: created by the author.

prototypes.

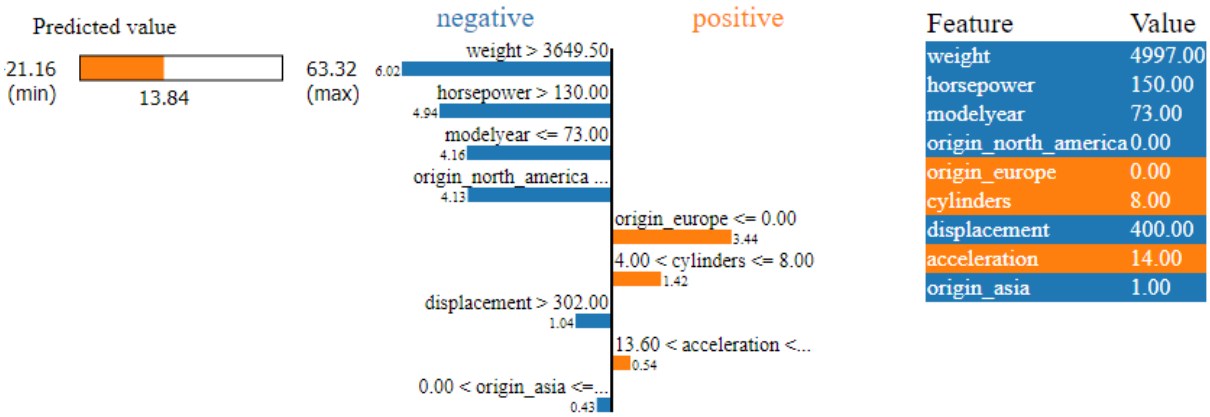
A.14.2 Local explanation

We will show the local explanations for the low and high values instances selected for the Auto-MPG data set, as shown in Table A.2.

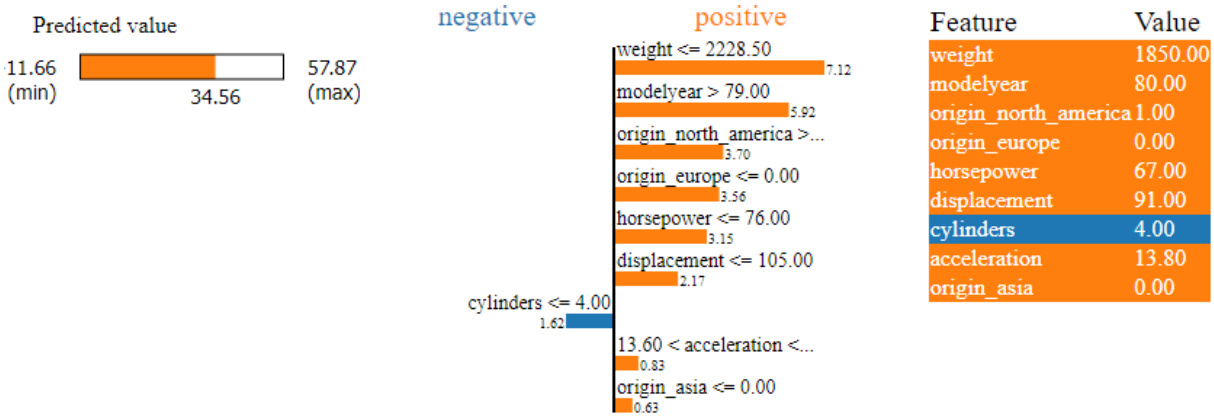
Fig. A.18 shows the SHAP local explanations for those selected instances. For the low-value instance, we can observe a similar contribution between weight, horsepower, model year, and displacement. They were responsible for decreasing the base value to the predicted value. On the other hand, a high cylinder helped increase the base value prediction. For the high-value instance, we can see that the lower weight had a higher contribution to increasing the value away from the baseline.

The LIME local explanation (Fig. A.19) resembles the explanation of a linear model combined with a decision tree. This local approximation creates additional information that can be extracted from the models for that small region around the instances of interest. For the low-value instance, we can see that weight above 3.6 tons will decrease the MPG by 6 points within this neighborhood. Although simplified by the discretization

Figure A.19: Kernel Regression local explanations using the LIME explainer for the Auto-MPG data set.



(a) Local explanation for the lowest target value



(b) Local explanation for the highest target value

Source: created by the author.

and limited to the neighborhood, this information can help readily identify the next steps. For example, when investigating the explanation for the high-value instance, we can see that 4 cylinders have a negative effect on the prediction, we could simulate a car with the same characteristics and a higher cylinder.

The ELA method returns as a local explanation for each instance the parameters of the linear regression performed with the five nearest neighbors (predicted value, coefficients, and intercept), the local importance of each of the features, and the local intervals in which each feature could be changed so that an instance with similar output was created, results shown in Table A.10. For the low MPG instance, ELA shows us that the most important features would be the model year (51.9%), weight (25.2%), and acceleration (19.2%). As for the high-value MPG instance, the most relevant features are weight (43.4%), model year (33.5%), and displacement (10%). As we usually have heavier cars when we are evaluating those with lower MPG, and since we know that the acceleration of a body is inversely proportional to its mass, this justifies that for low MPG vehicles, the importance of weight and acceleration are closer than when we are evaluating high

Table A.10: Kernel Regression local explanations using the ELA explainer for the Auto-MPG data set.

Low output	**High output**
y' = 12.9791	y' = 34.9871
Coefficients: [0, -0.1010, -0.0298, 0.0350, -2.5567, -2.2505, 0, 0, 0]	Coefficients: [0, -0.1633, -0.2393, 0.0304, -0.3278, 0.7370, -0.0438, 0.0654, -0.0216]
intercept: 83.1600	intercept: -44.8290
Importance of features:	Importance of features:
cylinders : 0.0	cylinders : 0.0
displacement : 3.137	displacement : 10.013
horsepower : 0.515	horsepower : 8.169
weight : 25.239	weight : 43.361
acceleration : 19.235	acceleration : 4.87
model year : 51.874	model year : 33.546
origin asia: 0.0	origin asia: 0.0
origin europe: 0.0	origin europe: 0.041
origin north america: 0.0	origin north america: 0.0
Local variation: 11.25 - 13.438	Local variation: 35.953 - 35.953
cylinders = 8.0	cylinders = 4.0
400.0 <= displacement <= 455.0	89.0 <= displacement <= 91.0
150.0 <= horsepower <= 225.0	62.0 <= horsepower <= 67.0
4746.0 <= weight <= 4997.0	1845.0 <= weight <= 1850.0
11.0 <= acceleration <= 14.0	13.8 <= acceleration <= 15.3
71.0 <= model year <= 73.0	model year = 80.0
origin asia = 1.0	origin asia = 0.0
origin europe = 0.0	2.0 <= origin europe <= 1.0
origin north america = 0.0	2.0 <= origin north america <= 1.0

MPG automobiles. Regarding the construction of similar cars in terms of MPG consumption, Table A.10 shows us that vehicles with outputs close to 11.2 to 13.4 MPG can be built with features similar to the low MPG instance evaluated, and some variations are tolerated, such as the weight between 4746 and 4997.

Table A.11 shows the counterfactual explanations for the selected instances where we aimed at increasing the MPG for the low-value instance and decreasing it for the high-value instance. Regarding the low value, we can see that DICE achieved the desired target values by reducing the displacement, horsepower, and weight and increasing acceleration and model year. It identifies the features that should be changed together and try to make a minimal change. For the high-value instance, it used the same features to achieve the target, but, in one case, it also decreased the number of cylinders.

Table A.11: Kernel Regression local explanations using the DiCE counterfactual (CF) explanations for the Auto-MPG data set. Unmodified values are shown as an hypen.

	Cyl.	*Disp.*	*HP*	*Weight*	*Acc.*	*Model year*	*Origin*			*MPG*
							Asia	*EU*	*NA*	
Original	8	400	150	4997	14	73	1	0	0	14.312
CF 1	-	350	105	3725	19	81	-	-	-	24.578
CF 2	-	260	90	3420	22	79	-	-	-	21.968
Original	4	91	67	1850	13.8	80	0	0	1	34.542
CF 1	-	120	97	2489	15	74	-	-	-	23.542
CF 2	3	70	100	2420	12.5	-	-	-	-	23.550

A.15 Black-box *vs* Gray-box and White-box models

As already mentioned, the explanation methods used to explain the black-box model are model-agnostic, meaning they can also be applied to explain white and gray-box models. As such, the main advantage of explaining the black-box models with these tools is that the

Black-box models have as it's main advantage to capture the non-linearity of the system being modeled and, in some situations, the interaction of the features. The explanation methods try to explain these features somehow. The PDP plot, for example, can extract similar information on the behavior of the model when varying the values of a given feature. This plot still has the disadvantage of not accounting for interactions and correlations between features, which can be inspected from the analytical expression provided by the gray-box models.

SAGE can quantify the global importance of the features by summarizing the Shapley values of each data point of the training set. These values correspond to the contribution of each feature to deviate each prediction from a baseline value (usually the average). These values can give us a sense of how much each feature contributes to a prediction on average. However, the practitioner should know that the average case may not reflect a single point of interest.

For this purpose, ELA provides a more insightful plot summarizing the importance of each feature for different values of the target. These can help the practitioner to make more assertive decisions revolving around particular points of interest.

We should be careful, though, that both SAGE and ELA do not provide the same straightforward information as the Partial Effect in which we have the expected variation of the target when making a slightly change to a feature.

Straying away from the feature importance, ProtoDash searches for prototypical

points that can act as a representative of the training data. This can be particularly insightful as it can help the practitioner to focus on a small number of points of interest, but has the disadvantage of not considering the model itself, only the data set.

Another form of explanation analysed in this section is the local explaination, where the explanation information is target to a single instance. A popular approach is the Shapley values that, for a given instance, quantify how much each feature contributed to deviating that output from the baseline. In other words, we can see how much each feature influenced the value of a particular output.

LIME creates a local linear model around the instance of interest, making the local explanation similar to the linear regression. In this case, the values associated with each feature depict the expected variation in the output after a slightly change in the feature value.

Finally, ELA as a local explanation model returns an information similar to that provided by LIME but in the form of a full report depicting not only the linear coefficients but also the variation in prediction between the black-box model and the linear approximation and the interval of values for each feature used to create the approximation.

In sum, for the global explanations, a combination of the PDP and ELA can return complementary insights about the model that can lead to further investigations. For the local explanations, the same thing can be said about SHAP and ELA, which returns different views of the prediction of a single instance.

Appendix B

Installation of the evaluated explanation methods

The Table B.1 describes how each explanatory method used in this work can be installed into the *Python* environment. We picked popular open-source explanatory methods based on different approaches to include in this survey, with online free available implementation.

Table B.1: Installation of explanation methods.

Explainer method	**Installation**
SHAP	`pip install shap` `conda install -c conda-forge shap` (alternative to pip) Note: In our experiments we are using version 0.41.0
LIME	`pip install lime`
ELA	Implementation available at https://github.com/renatomir/ELA-WCCI2020
SAGE	`pip install sage-importance`
Partial Effects	Implementation for regression available at https://github.com/gAldeia/iirsBenchmark

ICE/PDP	Available as a part of Scikit-Learn v1.0.0 or above:
	`pip install sklearn==1.0.0`
	`from sklearn.inspection import PartialDependenceDisplay`

ALE	`pip install PyALE`
ProtoDash	`pip install aix360`
DiCE	`pip install dice-ml`

Appendix C

Mental health assessment questionnaires: WHOQOL-BREF and BSI.

The WHOQOL-BREF questionnaire measures an individual's quality of life across four domains: physical, psychological, social relationships, and environment. Each domain is composed of a set of questions that are scored on a scale from 1 to 5, with higher scores indicating better quality of life.

These are the test questions:

1. How would you rate your quality of life?
2. How satisfied are you with your health?
3. To what extent do you feel that physical pain prevents you from doing what you need to do?
4. How much do you need any medical treatment to function in your daily life?
5. How much do you enjoy life?
6. To what extent do you feel your life to be meaningful?
7. How well are you able to concentrate?
8. How safe do you feel in your daily life?
9. How healthy is your physical environment?
10. Do you have enough energy for everyday life?
11. Are you able to accept your bodily appearance?
12. Have you enough money to meet your needs?
13. How available to you is the information that you need in your day-to-day life?

14. To what extent do you have the opportunity for leisure activities?
15. How well are you able to get around?
16. How satisfied are you with your sleep?
17. How satisfied are you with your ability to perform your daily living activities?
18. How satisfied are you with your capacity for work?
19. How satisfied are you with yourself?
20. How satisfied are you with your personal relationships?
21. How satisfied are you with your sex life?
22. How satisfied are you with the support you get from your friends?
23. How satisfied are you with the conditions of your living place?
24. How satisfied are you with your access to health services?
25. How satisfied are you with your transport?
26. How often do you have negative feelings such as blue mood, despair, anxiety, depression?

Questions number 1 and 2 are about general quality of life. The physical domain consists of seven questions, which are indexed as follows: 3, 4, 10, 15, 16, 17, and 18. The psychological domain consists of six questions, which are indexed as follows: 5, 6, 7, 11, 19, and 26. The social relationships domain consists of three questions, which are indexed as follows: 20, 21, and 22. Finally, the environment domain consists of eight questions, which are indexed as follows: 8, 9, 12, 13, 14, 23, 24, and 25.

The Brief Symptom Inventory (BSI) measures psychological distress across several dimensions, including somatization, obsessive-compulsive, interpersonal sensitivity, depression, and anxiety. The BSI consists of 53 items, with higher scores indicating greater levels of psychological distress.

These are the test questions:

1. Nervousness or shakiness inside;
2. Faintness or dizziness;
3. The idea that someone else can control your thoughts;
4. Feeling others are to blame for most of your troubles;

5. Trouble remembering things;
6. Feeling easily annoyed or irritated;
7. Pains in the heart or chest;
8. Feeling afraid in open spaces;
9. Thoughts of ending your life;
10. Feeling that most people cannot be trusted;
11. Poor appetite;
12. Suddenly scared for no reason;
13. Temper outbursts that you could not control;
14. Feeling lonely even when you are with people;
15. Feeling blocked in getting things done;
16. Feeling lonely;
17. Feeling blue;
18. Feeling no interest in things;
19. Feeling fearful;
20. Your feelings being easily hurt;
21. Feeling that people are unfriendly or dislike you;
22. Feeling inferior to others;
23. Nausea or upset stomach;
24. Feeling that you are watched or talked about by others;
25. Trouble falling asleep;
26. Having to check and double check what you do;
27. Difficulty making decisions;
28. Feeling afraid to travel on buses, subways, or trains;
29. Trouble getting your breath;
30. Hot or cold spells;

31. Having to avoid certain things, places, or activities because they frighten you;
32. Your mind going blank;
33. Numbness or tingling in parts of your body;
34. The idea that you should be punished for your sins;
35. Feeling hopeless about the future;
36. Trouble concentrating;
37. Feeling weak in parts of your body;
38. Feeling tense or keyed up;
39. Thoughts of death or dying;
40. Having urges to beat, injure, or harm someone;
41. Having urges to break or smash things;
42. Feeling very self-conscious with others;
43. Feeling uneasy in crowds;
44. Never feeling close to another person;
45. Spells of terror or panic;
46. Getting into frequent arguments;
47. Feeling nervous when you are left alone;
48. Others not giving you proper credit for your achievements;
49. Feeling so restless you couldn't sit still;
50. Feelings of worthlessness;
51. Feeling that people will take advantage of you if you let them;
52. Feeling of guilt; and
53. The idea that something is wrong with your mind.

Items 11, 25, 39, and 52 do not factor into any of the dimensions, but are included because they are clinically important. Specifically, the somatization dimension is composed of questions with indices 2, 7, 23, 29, 30, 33, and 37.