RODRIGO MONTEIRO DE AQUINO

NATURAL LANGUAGE EXPLANATIONS OF CLASSIFIER BEHAVIOR

São Paulo
2020

RODRIGO MONTEIRO DE AQUINO

NATURAL LANGUAGE EXPLANATIONS OF CLASSIFIER BEHAVIOR

VERSÃO CORRIGIDA

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Ciências.

São Paulo
2020

RODRIGO MONTEIRO DE AQUINO

NATURAL LANGUAGE EXPLANATIONS OF CLASSIFIER BEHAVIOR

VERSÃO CORRIGIDA

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Ciências.

Área de Concentração:
Engenharia de Computação

Orientador:
Fabio Gagliardi Cozman

São Paulo
2020

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 29 de Setembro de 2020

Assinatura do autor: Rodrigo Monteiro

Assinatura do orientador: _____

Catalogação-na-publicação

Aquino, Rodrigo
Natural Language Explanation for Classifier Behavior / R. Aquino -- versão corr. -- São Paulo, 2020.
69 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Aprendizado de máquinas 2.Interpretabilidade 3.Transparência I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

ACKNOWLEDGMENTS

I would like to thank Itaú Unibanco S.A. for supporting this work through the Itaú Scholarship Program.

Also, I thank PCS, and all the people at Escola Politécnica, for supporting my academic development. I would like to specially thank my advisor Fabio Cozman who guided me through all the research and development process in this work, and in the end helped me to be a better researcher.

To my colleagues Arthur Gusmão, Andrey Ruschel, Francisco Caio, Gabriela Melo, Roberto Fray and Gustavo Polleti who helped me overcome many problems that emerged during the course of the project. In each conversation with them I was able to learn a little bit more about several topics.

To my dearest friend Luiz Durão who inspired me to begin my master degree and helped whenever I needed assistance.

Finally, I thank all my family that has always been there for me, in special my mother, and my life partner Carolina Machado. Without them I wouldn't have made this far.

“I have been impressed with the urgency of doing. Knowing is not enough; we must apply. Being willing is not enough; we must do.”

-Leonardo da Vinci-

RESUMO

Enquanto modelos de aprendizado estatístico avançam em um número cada vez maior de aplicações reais, tem-se percebido que o entendimento das predições apresentadas por estes modelos é bastante desafiador. O campo de estudo focado em interpretabilidade/explicabilidade de inteligências artificiais tem desenvolvido diversas abordagens e ferramentas para melhorar o entendimento desses sistemas. Tais ferramentas tendem a ser direcionadas a cientistas de dados com conhecimento técnico. Os resultados obtidos a partir delas podem ser tabelas, gráficos ou outra representação gráfica (como superposição de cores em um texto, por exemplo); desta maneira, o usuário necessita de conhecimento técnico prévio para o consumo desta informação. Neste trabalho são implementadas técnicas que geram explicações textuais sobre o funcionamento interno de um dado classificador, focando em usuários com menor proeficiência técnica a respeito dos recursos de aprendizado de máquinas. Um pacote de geração de explicações textuais, chamado NaLax, foi construído e testado do usuários. Resultados preliminares foram publicados e apresentados na *IEEE International Conference of Artificial Intelligence and Knowledge Engineering* (AIKE) em 2019.

Palavras-Chave – Aprendizado de máquinas, interpretabilidade, transparência.

ABSTRACT

As machine learning models are increasingly used in a wide range of applications, there is growing concern about the challenges involved in understanding their predictions. The field of interpretability/explainability of artificial intelligences has developed several approaches and tools that aim at improving the understanding of such systems. These tools tend to focus on the knowledgeable data scientist as their main user. The tools usually produce plots, charts or another graphical representations (such as superposition of color on an image or text); thus the user must have some technical background so as to consume the information. This work developed techniques that generate a textual explanation for the internal behavior of a given classifier, aiming at users of machine learning with limited technical proficiency. A package for textual explanation generation, called NaLax, was built and tested with users. Preliminary results were published and presented at the *IEEE International Conference of Artificial Intelligence and Knowledge Engineering* (AIKE) in 2019.

Keywords – Machine learning, Interpretability, Transparency.

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

ABBREVIATIONS

AI	Artificial Intelligence
AIKE	Artificial Intelligence and Knowledge Engineering
ALE	Accumulated Local Effects
BLEU	Bilingual Evaluation Understudy
CEP-HU/USP	Comitê de Ética em Pesquisa do Hospital Universitário da USP
DARPA	Defense Advanced Research Projects Agency
EA	Explanation Agent
GDPR	General Data Protection Regulation
HCI	Human Computer Interaction
ICE	Individual Conditional Expectation
LDCP	Linguistic Description of Complex Phenomena'
LIME	Local Interpretable Model-agnostic Explanations
NaLax	Natural Language Explanations
NLG	Natural Language Generation
NLP	Natural Language Processing
PD	Partial Dependence
PDP	Partial Dependence Plot
TF-IDF	Term Frequency-Inverse Document Frequency
USP	Universidade de São Paulo
XAI	Explainable Artificial Intelligence

CONTENTS

1 INTRODUCTION AND MOTIVATION

The machine learning community has dedicated significant effort to develop techniques that interpret black-box classifiers such as deep neural networks (DARPA-BAA-16-53, 2016; RIBEIRO, 2016). As complex classifiers meet widespread application, it is important to make them understandable to a broader array of people. A model that is taken to be transparent may lack interpretability if its complexity exceeds a certain threshold. For instance, it may be difficult even to understand a logistic regression when facing a problem with a large number of features (more than 100, for example).

On top of that, recent legal issues have reinforced the need to explain decisions taken autonomously by complex classifiers. For instance, the *General Data Protection Regulation* (European Union, 2016), currently in force in the European Union (EU), and the law for personal data protection (*Lei Geral de Proteção de Dados* - LGPD), recently implemented in Brazil, require automatic decisions to be explained if so requested.

The need for a "less opaque" view of learned models is a natural concern for several business sectors in which highly complex models have been widely used. For instance, macro-level risks may increase with the lack of interpretability in applications in finance (HAGRAS, 2018) and medicine (HOLZINGER, 2017), in which the decisions made have serious impact (social and economic). Hence the professional worker in the field must understand how and why a model yields each possible output.

A recent study by Miller (2019) points out that the discussion of explanation in artificial intelligence is an old topic, and even though several authors have worked towards a formal definition of what an explanation is, there is no final consensus on a definition. Even though there is no consensus in the literature on what is the formal definition of interpretability, this work adopts the definition by Doshi-Velez: "the ability to explain or to present in understandable terms to a human" (DOSHI-VELEZ; KIM, 2017).

Following the terms defined by Miller (2019), the construction of an explanation may often be taken as the answer to a "why-question". This type of question, as detailed by the author, is a

combination of another 2 structures: a “whether-question” preceded by the word “why” and the presupposition that the event in the question has occurred. In other words, one have to construct a sentence in which could result on a **yes** or **no** answer and add the word “why” in front of it meaning to question an previously occurred event.

Several research results have appeared on interpreting and explaining classifiers (RIBEIRO, 2016; MONTAVON, 2017; GOLDSTEIN, 2015). An entire program led by the Defense Advanced Research Projects Agency (DARPA) started in 2016, focused on developing new approaches that have explanations built in their core. The Explainable Artificial Intelligence (XAI) program (DARPA-BAA-16-53, 2016) aims to build effective interfaces to present the explanations to end users. The envisioned new interface is to be based on state-of-the-art Human Computer Interaction (HCI) technologies, and to be able to present explanations based on analogies with visual and textual means of communication. The psychology of explanation is emphasized in DARPAS’s program, as depicted at Figure 1.1. The user’s background and knowledge have to be taken in consideration when formatting an explanation within a given context.

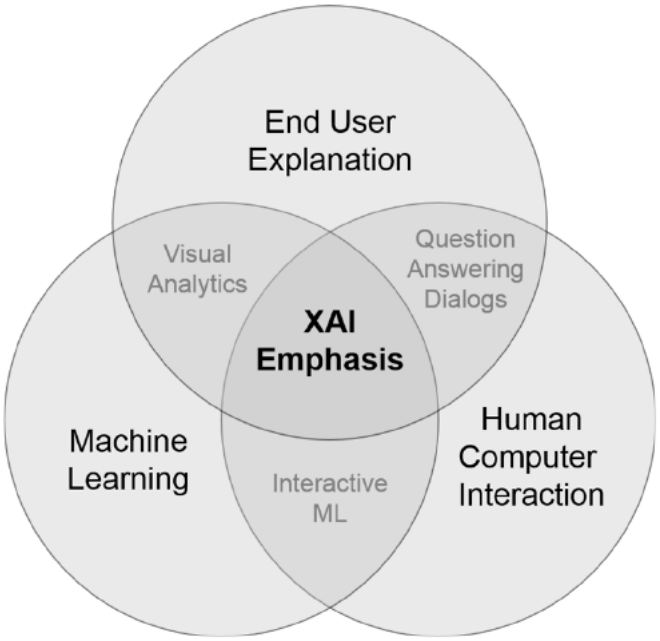


Figure 1.1: XAI program areas (source: DARPA-BAA-16-53 (2016)).

Indeed, the best way to explain a classifier depends on the end user: a data scientist may be very happy with a linear equation and a few graphs relating weights to outputs, while a less proficient user, say a member of a legal team, a human resources professional, or a final customer, may be uncomfortable when presented only a mathematical explanation. In particular, auditing bodies who are overseeing whether automatic classifiers can be properly interpreted may benefit from explanations that do not require advanced mathematical expertise.

That is, a system that wishes to ensure interpretability has to adapt its presentation layer to its users. For instance, a system that is used on a hospital to assist a surgeon will have different requirements compared to one supporting the decision of a doctor examining a patient. It is naive to expect that interpretability should be addressed solely by the perspective of a machine learning practitioner.

Several applications may benefit not only from the understanding of a model inner workings, but from a human-readable rationale behind outputs. For instance, in the system by Vlek, Prakken, Renooij and Verheij (2016), an explanatory text is generated from a Bayesian network applied to law. In their application (Section 2.2), interpretability was not pursued to improve the trustfulness of the model, but to illuminate its reasoning.

Implementation-wise, generating a textual explanation from a model is a similar to generating text from an expert system or from any other data (the generation of weather forecasts based on data is an example (REITER; DALE, 1997)). The high level processes are illustrated in Figure 1.2.

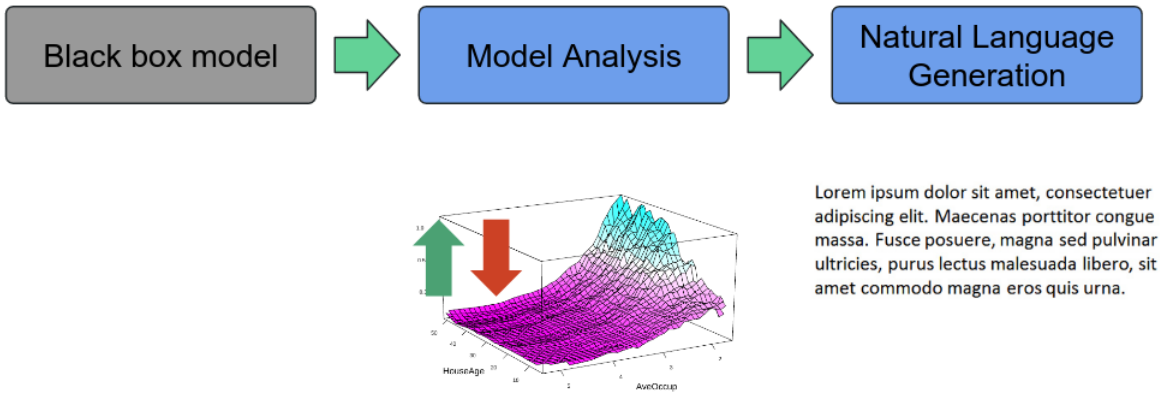


Figure 1.2: Generating text from a model (source: author (2020)).

This work implements a set of techniques that emphasize textual explanations; the goal is to generate a readable explanation for the behavior of a given (complex) classifier. The explanation is not expected to depend on the design of the model; rather, the explanation captures the overall behavior of the model without any attempt to justify the ultimate causes of a classification. In particular we aim at users with some mathematical sophistication but no serious knowledge of data science — the kind of user we anticipate to see in auditing and regulating bodies.

Given that the definition of a "good explanation" is highly dependent on the listener, some subjective metrics have to be applied to measure the quality of an explanation. The DARPA program describes metrics that can measure the effectiveness of an explanation; user satisfaction is taken as the top priority at their expected evaluation sequence, so explanations have to be

clear and useful to the end user. In this work, we applied established metrics to evaluate our framework as compared to others in the literature. Our experiments, detailed at Sections 4.1.3 and 4.2.2, tried to capture the subjective perception of interviewees reacting to explanations, particular with respect to clarity and trustworthiness. We examined explanations related to the Wine Dataset (a well-established literature baseline dataset to evaluate models), and then run an experiment focused on predictions related to the COVID-19 pandemic; in the latter case an interpretable model is important for users.

A paper (AQUINO; COZMAN, 2019) describing part of this work was accepted the IEEE International Conference of Artificial Intelligence and Knowledge Engineering (AIKE) in 2019, which reflected the relevance of this work in the scientific community.

This dissertation is organized as follows. Chapter 2 discusses related work and applications where interpretability is applied (not restricted to text generation). Chapter 3 presents our approach to generate textual explanations using state-of-the art techniques (PDP in Section 3.1.1 and ICE plots in Section 3.1.2); that chapter also describes an evaluation technique (Section 3.1.3) and natural language generation (Section 3.2). Chapter 4 presents an experimental validation study applied to two relevant datasets, with feedback from users. The main goal was to verify the correct generation of the outputs and the usability of the approach in comparison to other well established tools. Chapter 5 presents the final comments on this work.

2 RELATED WORK ON INTERPRETABILITY

Even though an accepted formal definition of interpretability is still lacking, there is wide interest in making automatic classifications more interpretable to the human end user (DOSHI-VELEZ; KIM, 2017). Improvements in interpretability have become urgent due to the application of machine learning techniques in several fields, targeting a variety of users.

The social sciences have already extensively studied the social meaning of explanations, as reviewed by Miller (2019). He exposes work by psychologists and scientists that have explored how people build their explanations. In particular Miller presents distinctions between the terms interpretability, explainability, and justification. Explainability is defined as the degree to which an observer can understand the cause of a decision (of a model) presented by an explaining agent. A justification instead points out the reasons why a presented decision is a good one (however it may not expose the inner decision making process that a model took to reach it). The present work focuses on the exposition of the inner workings of a model; it does not try to justify the decisions.

A few popular models employed in machine learning are taken to be intrinsically interpretable: for instance, simple logistic regressions and shallow decision trees (GUIDOTTI, 2018). A simple analysis of weights or a simple visual inspection are enough to reveal the behavior of these models. Non-interpretable models, often referred to as black box models, depend on complex structures; for instance, deep neural networks are quite hard to understand without proper tools. However, even models that are considered intrinsically interpretable may be almost impossible to interpret when they are applied to complex problems —- take deep decision trees or logistic regressions with many features.

Explanations may focus on *local* decisions (that is, they explain why a particular output was generated from a particular input) or they may offer a *global* view of a classifier's behavior (GUIDOTTI, 2018). A bank would use a local approach to explain to a client the reasons as to why credit approval was denied to him, however a regulatory agency may be interested in the overall behavior of the classifier to identify possible bias in it.

Some interpretability techniques are only applicable to specific models, such as neural networks; they are said to be *model-specific*. Techniques such as Layer-wise Relevance Propagation (LRP) (BACH, 2015) and deep Taylor decomposition (MONTAVON, 2017) are examples of model-specific techniques that apply to neural networks, since they rely on the specific architecture of the models in study. Since our approach focus on a more broad view of analysis these techniques are out of our scope of study.

Other techniques can in principle be applied to any model; they are said to be *model-agnostic*. The Local Interpretable Model-Agnostic Explainer (LIME) (RIBEIRO, 2016), Partial Dependence Plots (PDP) (HASTIE, 2009) and Individual Conditional Expectations (ICE) (GOLDSTEIN, 2015) are examples of model-agnostic techniques. The PDP and ICE techniques are further detailed in Sections 3.1.1 and 3.1.2 respectively, as they are important later in this work.

A key point about these previous techniques, and indeed several others in the literature (HECHTLINGER, 2016; PURI, 2017), is that they formulate their explanations focusing a data scientist as the reader. That is, they assume a rather sophisticated user, as they generate reports based on plots, charts and other elements that are familiar to those already working with data science.

In the remainder of this section we review a few relevant proposals in the literature, as they stand for several key strategies concerning explanation generation. More specifically, they are applied to real use cases, where interpretability is necessary for model trustfullness.

2.1 LIME

The approach proposed by Ribeiro, Singh and Guestrin (2016) in LIME was to create a surrogate (and easily interpretable) model that locally approximates a given complex model, guaranteeing that the former is faithful to the latter on the surroundings of an specific decision. As an interpretable model can be used to approximate locally any black box, the resulting method is agnostic to any type of classifier.

To learn the local behavior of the complex model, LIME generates new samples around the sample provided by its user (these new samples are uniformly generated). In Figure 2.1, the dashed line depicts the simple model fitted by LIME as an interpretable representative of the complex model around a particular point (the red cross). In this figure we see that the dashed line does separate blue spots and purple crosses around the point of interest; there is a faithful representation for the complex model near the provided sample.

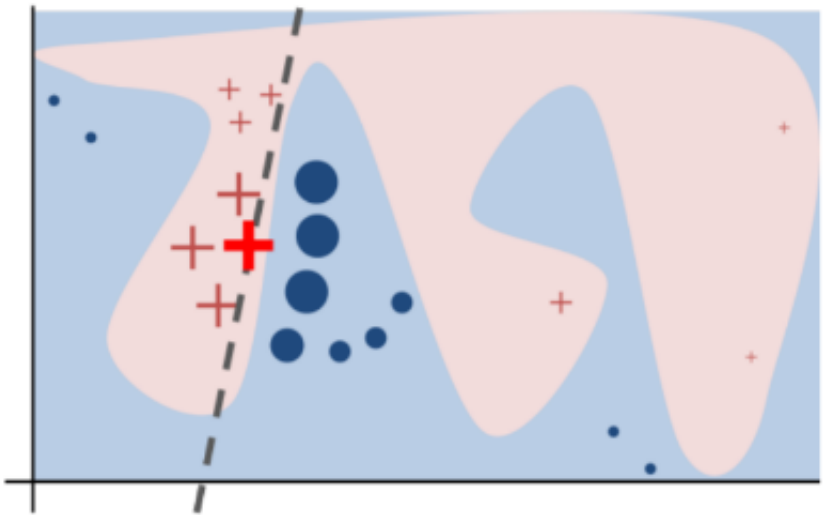


Figure 2.1: Representation of a local approximation performed by LIME (source: Ribeiro, Singh and Guestrin (2016)).

The publicly available distribution of LIME implements a logistic regression. Figure 2.2 (RIBEIRO, 2016) shows an explanation given by LIME for a particular prediction, where the model predicts the chance of default for a client (the dataset from which this model was learned stored historical payment information of the client and some of its personal information).

The first part of this figure depicts the chance of each outcome (not default has 85% probability). The second part exposes how each feature contributes to the prediction (positively or negatively with respect to each possible class of prediction). The last part shows the values produced by LIME. Such visual information can be used to explore different scenario around the given samples, possibly answering questions such as: "How does the probability of default vary when a client delays one payment?"

Note that the representation in Figure 2.2 can be used for any data in tabular form. For

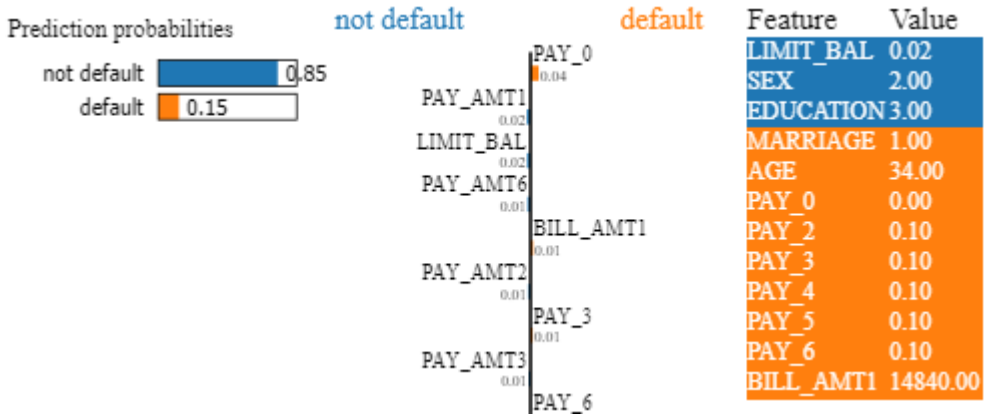


Figure 2.2: Example of a LIME explanation (source: Ribeiro, Singh and Guestrin (2016)).

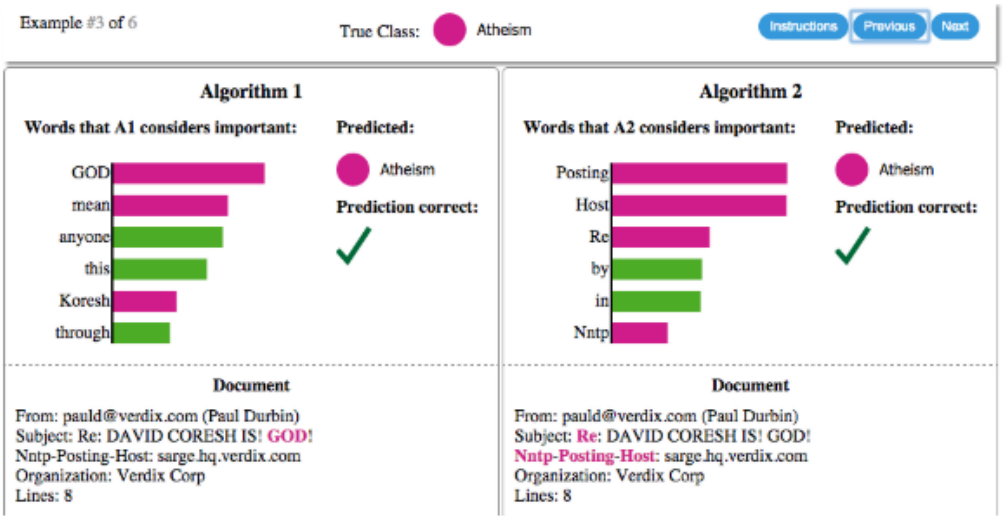
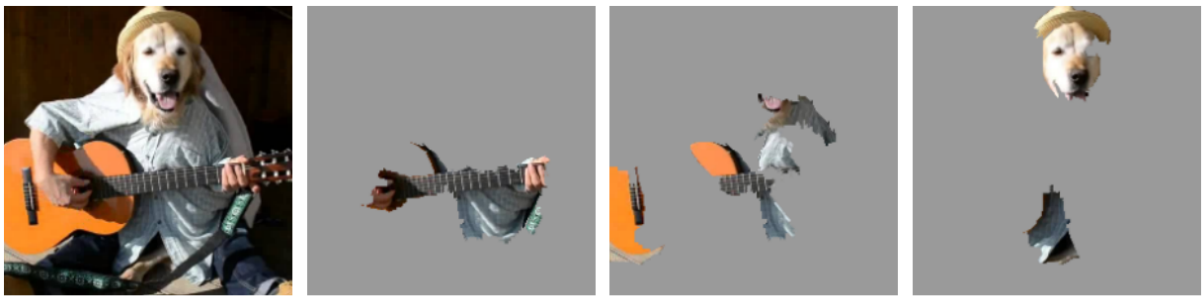


Figure 2.3: Example of a LIME explanation for text classification (source: Ribeiro, Singh and Guestrin (2016)).

textual data, however, this representation is not easy to process. One usually converts texts into vectors so as to fit a model, and in this process the meaning of features and how they influence the decision of a classifier is blurred. For these settings, LIME can produce an output that highlights the most important words in an actual text provided as sampled of analysis (Figure 2.3). A scale with the most important words that supported the output of the model is presented to the user.

Similarly, the interpretation of image classification follows a special scheme. LIME uses the concept of super-pixel (that expresses the presence or absence of a contiguous patch of similar pixels). The example in Figure 2.4 conveys the explanation for the top 3 classes predicted by the pre-trained neural network Inception. For each explanation the gray area covers the super-pixels in the image that do not justify the predicted class, exposing the remainder ones as the reason why it was predicted. LIME indicates which parts of the image led it to be labeled as an electric guitar (Figure 2.4b), as well as acoustic guitar and labrador (Figures 2.4c and 2.4d).



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 2.4: Example of a LIME explanation for image classification (source: Ribeiro, Singh and Guestrin (2016)).

These explanations enhance trust in the classifier when they indicate that it is not acting in an unreasonable way. If the explanation given by LIME does not display a reasonable connection between input and output, the user should consider improving the model.

2.2 Bayesian Network Explanations with Scenarios

In this section we review relevant work by Vlek, Prakken, Renooij and Verheij (2016). Their objective was to present textual arguments that supported a model decision in a criminal trial, so as to assist the jury and their judge in their decision. This work is detailed here since it relates to the current research as it is an approach to enhance interpretability of a model being used on a real application - in this case, a trial.

They adopted Bayesian networks to represent connections amongst variables as in Figure 2.5. In this application no learning method is being applied since the network is representing a possible scenario. Therefore, the relevance of this work relies on the enhancement of the interpretation of the model used and its feasibility towards all facts in it.

The quality of a scenario is measured by the probability of the outcome of interest. Given that in a trial there may be different plausible scenarios, and for any of them several elements (facts or proofs) intersect, the amount of time needed to evaluate this network of possibilities could be a problem in a time constrained situation.

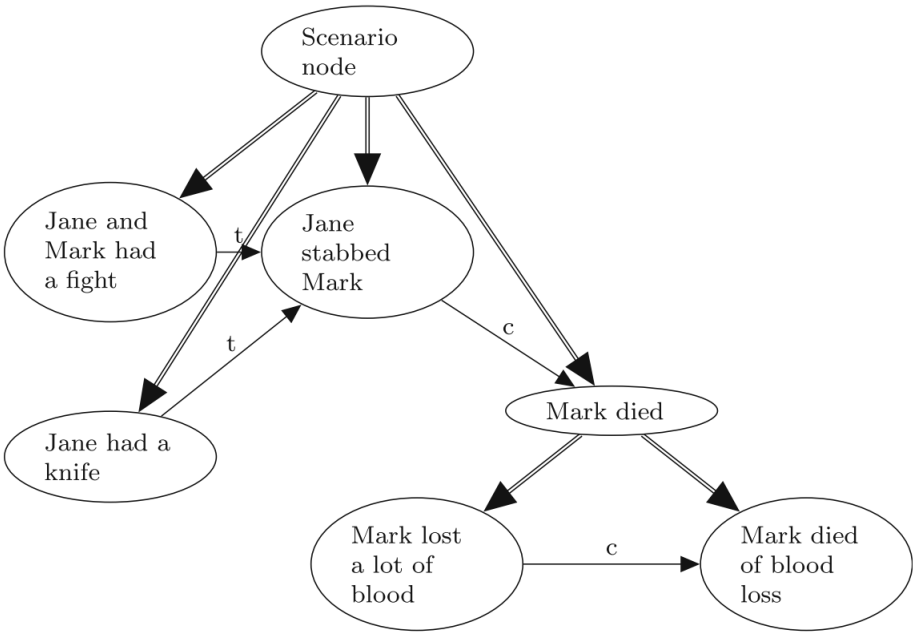


Figure 2.5: Example of structure with scenarios (source: Vlek, Prakken, Renooij and Verheij (2016)).

In their proposed representation, one can identify the temporal sequence of events using arrows with a label **t**. It is then possible to identify inconsistencies in a sequence of events. Similarly, causal relations can be identified with a **c** above the connective arrows. A scenario starts at a node with double lined arrows coming out. Depending of the starting node of the analysis, their approach may generate different plausible scenarios explanations. Here is an example:

> "Jane and Mark had a fight and Jane had a knife. Then Jane stabbed Mark. Therefore, [Mark died: Mark lost a lot of blood. Therefore, Mark died of blood loss]"(VLEK, 2016).

This textual explanation, extracted from one of the possible readings of the structure, describes the causal relation between the fact that Mark died and the blood loss. This information relates to the previously presented fact that Jane stabbed him.

With this tool in hand, jury and judge in a trial may have a clear view of all the possible interpretations of a case. The graphical representation by the Bayesian model, with the addition of these textual elements, provides a coherent interpretation that may lead to a high confidence verdict.

2.3 Linguistic Description of Complex Phenomena

A rather comprehensive approach to explanation generation is offered by the LDCP (Linguistic Description of Complex Phenomena) method (CONDE-CLEMENTE, 2017). The method is implemented in the rLDCP library for the R language.

LDCP generates textual reports based on tabular data; it uses fuzzy logic algorithms to determine appropriate levels of input values that are turned into textual form. LDCP also uses fuzzy logic to handle non-numeric or imprecise information, as well to deal with the inherent vagueness of natural language (NOVAK, 1999). The LDCP architecture is depicted in Figure 2.6. Each stage of the architecture is filled with prior knowledge of the phenomena of interest, and one can generate an automated report of the data originated from the phenomena.

The output of a model can be expressed as a textual response filling the gaps of a template to produce the desired report. This final step is the interface between LDCP and the user, and must be adjusted to the application of interest.

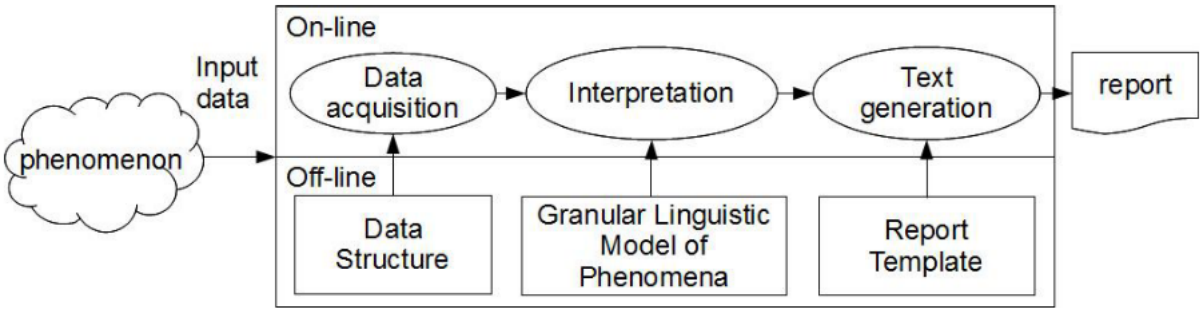


Figure 2.6: LDCP architeture (source: Novak, Perfilieva and Mockor (1999)).

2.4 Human-Centric Explanations of Predictions

Biran and Mckeown (2017) focus their work on "human-centered" explanations that are based on NLG using narrative roles. Their output is a brief text; Figure 2.7 shows an example explaining predictions given by a classifier. Figure 2.8 shows another piece of the explanation their system generates. The focus on their work is geared towards the identification of the most important features to present in the generated text. This is accomplished through a metric called narrative role.

The prediction is that this stock's price will **RISE**.

While there is strong evidence in ev-ebitda-ratio, which is normal, prior exhibits normal strong counter-evidence. Momentum_1d_11d and return-on-revenue provides unusually strong counter-evidence.

Return-on-assets provides unusually strong evidence. Furthermore, return-on-assets gives an indication of the capital intensity of the company, which will depend on the industry; companies that require large initial investments will generally have lower return-on-assets. Also, return-on-assets is one of the elements used in financial analysis using the Du Pont Identity. Moreover, return-on-assets is an indicator of how profitable a company is before leverage, and is compared with companies in the same industry. Return-on-assets is a common figure used for comparing performance of financial institutions (such as banks), because the majority of their assets will have a carrying value that is close to their actual market value. Return-on-assets is not useful for comparisons between industries because of factors of scale and peculiar capital requirements (such as reserve requirements in the insurance and banking industries).

Figure 2.7: Text output of Biran and Mckeown human-centric approach (source: Biran and Mckeown (2017)).

Even though their approach produces explanations that are similar to the ones we have developed in the work reported here, their work focused heavily on the selection of core messages to be presented to the user. We instead focus on a global explanation of a model's behavior,

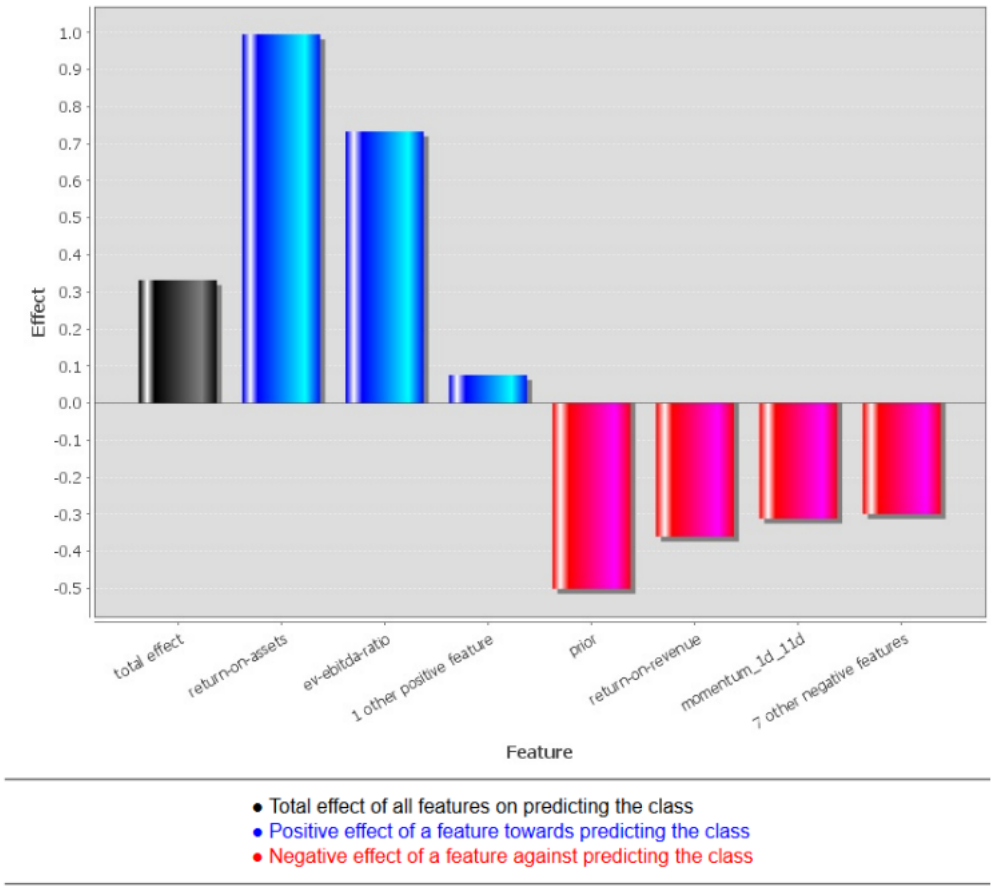


Figure 2.8: Chart output of Biran and Mckeown human-centric approach (source: Biran and Mckeown (2017)).

explaining the most common outputs generated by combinations of features.

In the end, we wish to provide readable, even if long, explanations to any person, regardless of its proficiency, of how a model works and how the inputs affect predictions. We wish to create full reports for the model of interest, creating a complete documentation – or "snapshot" – of the current state of the model. To do so we use ideas inspired in the works described in this section.

3 PROPOSED SOLUTION

The final goal of this project is to create an agnostic framework that generates textual explanations for a classifier. Even though we focus on global explanations, Section 3.1.4 discusses local explanations that employ the global analysis.

To achieve the intended global analysis, the output of a given model is analyzed as its inputs vary and the relationship between inputs and output is summarized in a textual report. There are techniques in the literature that aim at capturing input-output relationships; in particular we resort here to PDP (HASTIE, 2009) and ICE (GOLDSTEIN, 2015). Even though these tools are actually "plots", they are not actually drawn to the user in our approach; only the calculations behind PDPs are used to capture the interactions between features and how they influence the chance of a determined class to occur.

Miller (2019) points out that presenting dry statistical relationships to explain events is unsatisfying to human users; here we dynamically generate explanations by comparing odds of a range of values with odds of other ranges of values. The goal is to answer the following question: "Why did the model produce a particular classification?"

This chapter will detail each step of our framework; Figure 3.1 shows the high-level view of the framework. Except from the black box model, each one of the blocks will be discussed in the next sections.

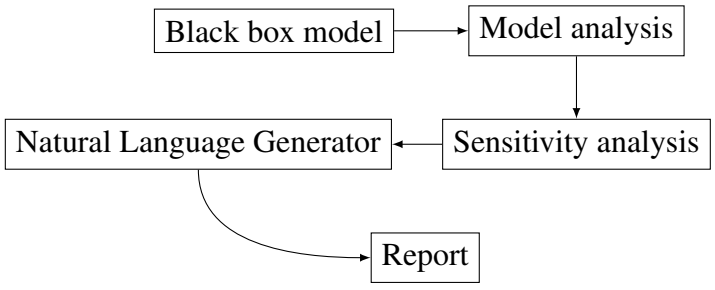


Figure 3.1: Steps in report generation (source: author (2020)).

3.1 Model analysis

As our work aims to be model-agnostic, we employed two techniques from the literature that do not depend on the particular model structure: Partial Dependence Plots and Individual Conditional Expectation. This section describes these techniques.

3.1.1 Partial Dependence

A Partial Dependence (PD) function (HASTIE, 2009) shows the marginal effect on the model's predicted outcome when we vary the values of a subset of the input features. Equation (3.1) defines a partial dependence function:

$$\bar{f}_S(X_S) = \frac{1}{N}\sum_{i=1}^{N} f(X_S, x_{iC}), \tag{3.1}$$

where $\bar{f}_S$ stands for the marginal average of the function defined by the model, X_S is a subset of interest of the input features, N is the total number of training samples, x_{iC} assumes values of the complement of X_S in the training data (values of the features that are not in the subset of the features of interest).

A PDP offers a graphical representation of such a function; as a PDP is limited to three dimensions for presentation purposes, it is restricted to two input features at a time. For a classifier model that produces probabilities of classes, the PD functions return the probability of a class to have a true value when the features of interest have certain values.

Equation (3.1) tells us that if we want to measure the model's behavior in respect to a set of features (X_s) then we must calculate the mean value of the model's response for all combinations of values for the other features (X_C) - calculated with each instance x_{iC}. This process may be computationally expensive as the algorithm computes each point using values of all samples in the dataset.

As an example, if we want to analyze the marginal influence of two features in the prediction of a model, for each combination of values of the two features of interest we must calculate the mean result of the predictions obtained by varying the other features within a certain range. Such range is usually defined by looking at the training data used to fit the model. The result of this calculation is a global view of the model's behavior.

Figure 3.2 illustrates a PDP for two features on the California housing dataset (KELLEY; BARRY, 1997), where each point represented in it reflects the mean result of the predictions

obtained when the features HouseAge and AveOccup have any pair of values on the axes.

One can see that, for feature AveOccup (average number of people living in the house), there is almost no variation of house price for values greater than 3. However, for values smaller than 3 the output depends heavily both on this feature and on the values of HouseAge.

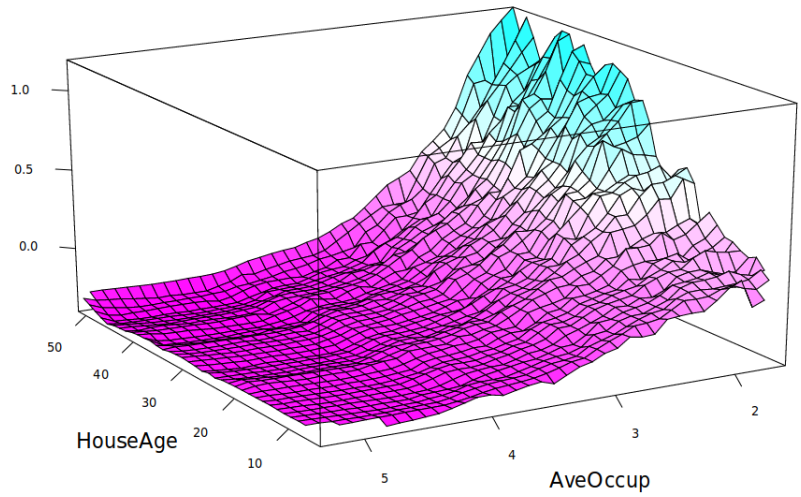


Figure 3.2: PDP of house value on median age and average occupancy (source: Hastie, Tibshirani and Friedman (2009)).

There is an obvious limitation of this method that is the maximum number of simultaneously analysed features being just 2; there is no visual way to add another dimension to represent more interactions amongst features. This limitation is not a problem in our approach as our goal is to generate textual explanations instead of graphs; the description of a model's behavior can be as complex as requested by the user, allowing control over the granularity of the reports.

We adopt the idea behind PDPs; that is, our the focus is to analyse the surfaces of probability as selected features vary. These surfaces, which can be high dimensional, capture trends of a particular class label to be selected under various circumstances. The final output expected from the PDP in our approach is a textual description of the surfaces generated for each class label in the model. With this data we can reason about the behavior of the model.

3.1.2 Individual Conditional Expectation

The Individual Conditional Expectation (ICE) technique, described by Goldstein, Kapelner, Bleich and Pitkin (2015), offers an alternative to PDPs. An ICE plot shows the output variation when a single feature of an individual instance varies. It has been shown that a PDP can be calculated from the average of curves generated by the ICE technique; hence both methods are related.

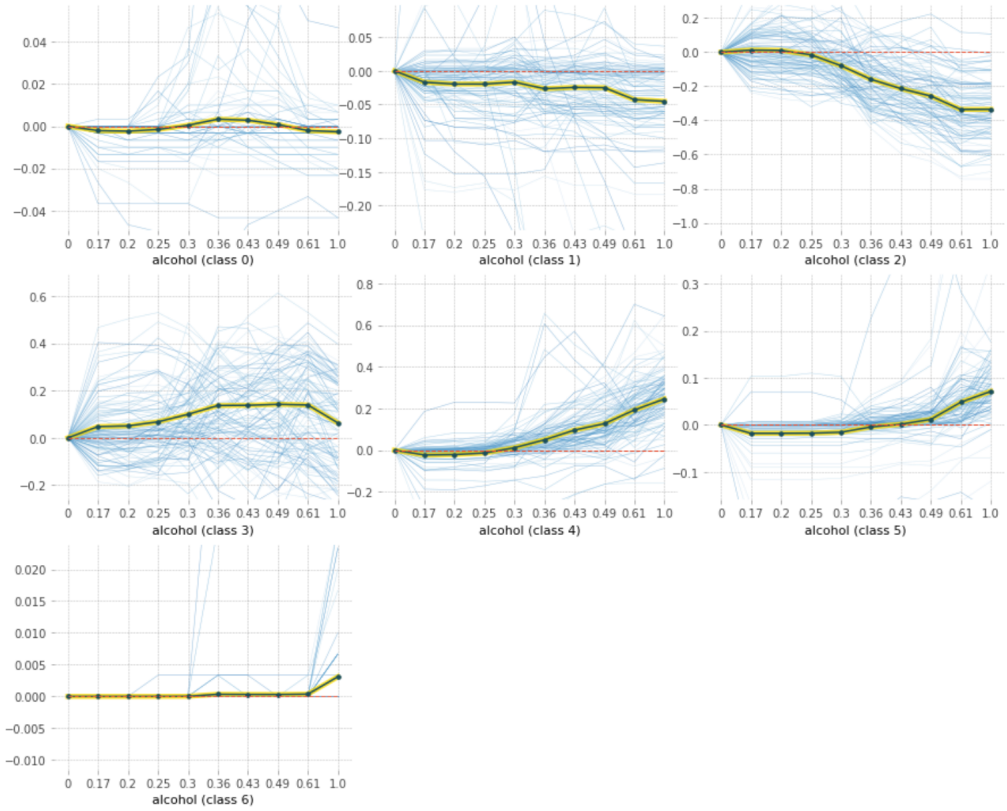


Figure 3.3: ICE plots for the Wine dataset, where each one represents one possible prediction label (source: author (2020)).

The Wine Dataset (CORTEZ, 2009)) is a well-established literature dataset to evaluate classification models. It has several features related to the biochemical components of two types of wine (red and white) and a feature that indicates the "quality" of the sampled wine. The models fitted with this dataset aims to predict the quality of the wine (ranking it with from 0 to 9 - named as classes in this work) based on its biochemical feature values.

For a multi-label problem, one plot is generated for each possible label, as depicted in Figure 3.3. In this figure an ICE plot is presented per label analysing the feature alcohol in the predictions given by a random forest classifier.

Note that it is possible to analyse each label in comparison to the others directly as the starting point is always zero.

Each plot presented in Figure 3.3 contains dozens of lines, each one capturing an instance of a prediction given by the classifier of interest. The mean value of all predictions is shown in each of these plots by the thickest line. This "mean curve" is the curve given by the PDP for the same feature. In a sense, the ICE curves capture the local interpretation of a classifier, something lost with the PD functions.

As discussed in Subsection 3.1.1, we wish to get information of how the probability of

a given label changes when some input features vary. With this in mind, as we had to alter the output of the PDP: we only get the data from each curve on each ICE plot to describe the interaction we are interested in. To generate descriptive information, we have to apply an aggregation method; as ICE produces several curves for each label, we use the model statistics to group similar curves, generating an aggregated view of the information shown in the chart. Figure 3.4 illustrates the characteristic of this technique, which generates many different curves for each label.

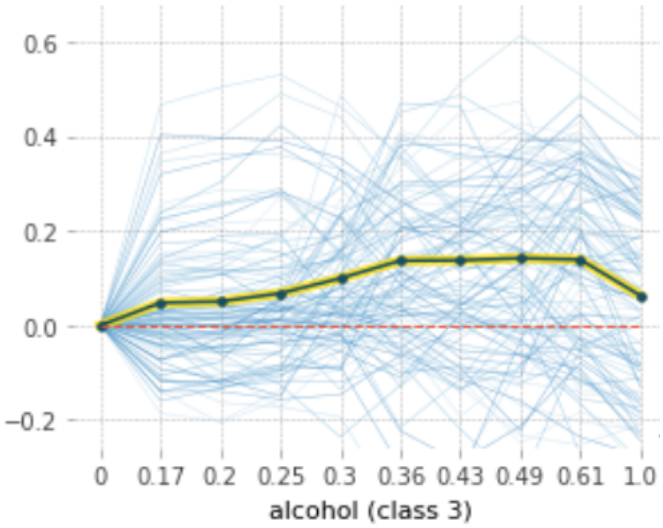


Figure 3.4: Detail of the ICE plot for class 3 (source: author (2020)).

3.1.3 Describing Partial Dependency functions with sectors

To study the results provided from the analysis of the model in the previous step in our approach, the points in space representing the PD function or the ICE curves are aggregated in sectors where its mean behavior is identified and summarized.

Our approach computes the PD function and the ICE curves for each possible label; from there, descriptive texts are generated describing how the gradient behaves throughout sectors in the probability surfaces. For such calculations we resort to standard methods of gradient computation (FORNBERG, 1988) as they are implemented with Numpy (a Python programming language module).

Given that it is unfeasible to describe a high-dimensional surface by detailing its behavior at every possible point, the setup depicted in Figure 3.5 shows an example of a sector that will have its gradient calculated (mean value for each point in it) and stored in a table with all of the other sectors' values.

The bounds of a sector is defined as a ratio of the input values of the features of interest It

is calculated based on a sensibility variable set by the user. With the ratio of the sectors defined, the mean gradient is calculated for each one - to summarize the tendency within it (increased or decreased chance of a prediction class to occur).

These steps, which are reproduced to each class given by the model, led the overall trend in each sector to be clearly exposed in the next stage in the analysis, natural language generation. The result is a set of surfaces described by several sectors and their mean gradient values.

The visualization of sectors in a three dimensional space (PDP with two input features) is shown in Figure 3.5: two features of interest have their range of values determined and the PD function is calculated to generate the value on the vertical axis (that is, the average of the probability of the label of interest over all possible values of the non-fixed features).

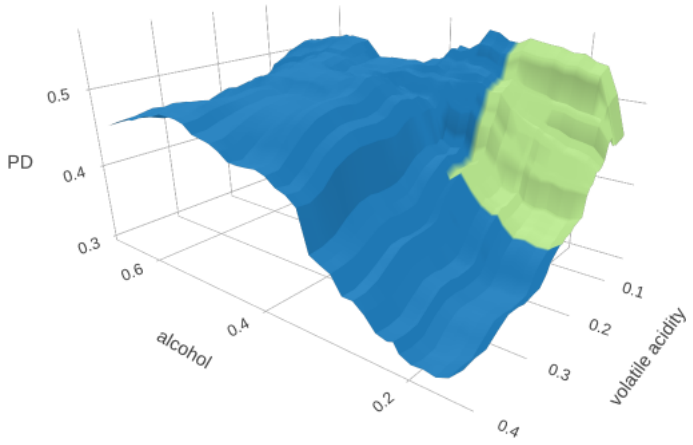


Figure 3.5: Highlight of a sectors on a PDP (source: author (2020)).

Algorithm 1, in Appendix A, translates the steps described before into code detailing how the sectors in the PD function surface are determined and how the mean gradient is calculated in each one of them. Steps in that algorithm are repeated for each label, limited only by a *sensitivity* value that bounds the effect of less relevant labels and controls the size of the sectors. The computed gradient is averaged in each sector to determine a trend; this information feeds the natural language generation step.

3.1.4 Local explanation: in-sector sample placement

As the plots offered by PDPs and ICE cover the whole space of combinations for input and output, the result they generate is a global analysis of the model. However, there are scenarios where one may be interested in a local explanation (that is, the explanation for a classification). The methods can be adapted so as to produce an approximate local explanation. This approximation comes from the basic algorithm of the PD functions, which calculates mean values over

a series of predictions given by a model. The values of a PD function describe a surface on a N-dimensional space (N being the number of features of study plus 1) which is segmented on several sectors. Any sample data of interest can be placed in this space, and identified in which sector it resides (as depicted in Figure 3.6).

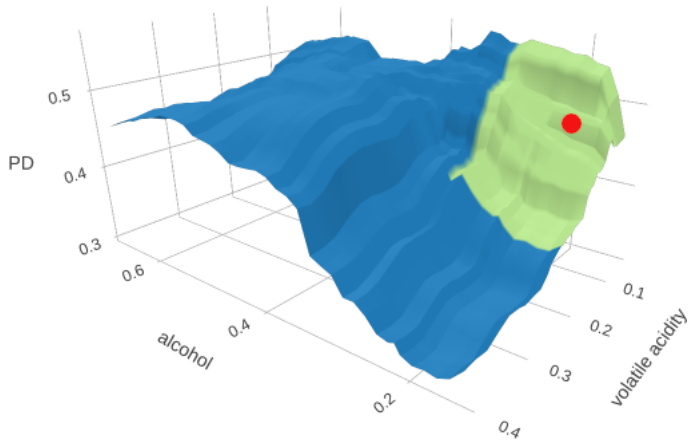


Figure 3.6: Sample placement in a sector (source: author (2020)).

So, to advance a bit over topics to be discussed in Section 3.2, a local explanation for a given point could be:

> The provided sample falls in the sector defined by the interval of values 0.1 and 0.2 of feature Volatile Acidity and values 0.1 and 0.2 of feature Alcohol. This sector has a slight decrease in the probability for the predicted class 6 to occur.

Note that this local analysis is made relative to the features studied in a previous global analysis, in which a number of features are selected by the user to calculate the PD function. If one wishes a take more features into account in the local analysis, it is necessary first to apply the PD calculations to more features (thus increasing the dimensionality of the resulting representation).

3.2 Natural Language Generation

This section describes techniques that we used to produce textual explanations. The field of Natural Language Processing (NLP) covers a vast range of different application, such as machine translation (WOŁK; MARASEK, 2015), sentiment analysis (DOS SANTOS; GATTI, 2014), text summarization (NAZARI; MAHDAVI, 2018).

The present work employed concepts of Natural Language Generation, in particular based

on the architecture detailed by Reiter and Dale (1997), who thoroughly describes steps to generate text from data (Figure 3.7).

The Text Planner transforms the input data into structures that will later generate the desired message to the target user. The structure of the output sentences are engineered in the Sentence Planner, which selects words that will represent information in the final text. This module may have some intersection with the previous one. Finally, the Linguistic Realiser constructs the sentences, according to the structures built in the previous modules and respecting the grammatical rules of the target language.

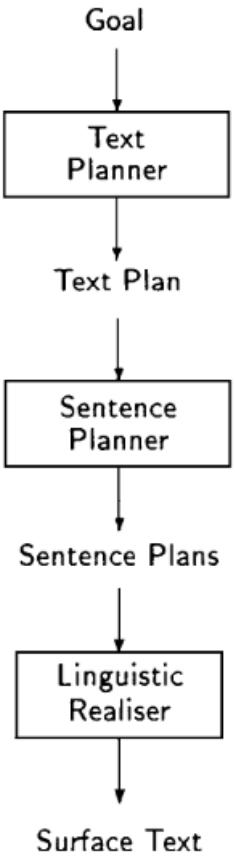


Figure 3.7: Full architecture of a NLG system (source: Reiter and Dale (1997)).

The system implemented in this work followed this architecture; however the code did not follow the strict moularization described by Reiter and Dale (1997). The remainder of this section discusses the rationale and the implementation of our system.

Aiming to generate a text that could be easily interpreted by any professional, this module is built to mimic the way a person reads a chart. Usually, the description of a chart focuses on the rises and falls of a function in some selected ranges of the axis presented.

In natural language, one might say:

"There is a ① high ② increase in the probability of label y when feature X varies from a_1 to a_n and feature Z varies from b_1 to b_n"

This sentence can be broken down into the following elements:

- **trend indicator**

  Term that indicates whether there is an increase, decrease or no change in the trend of the analysed plot. It reflects the signal of the gradient calculated in the sector (positive/increase, negative/decrease and near zero/ no tendency). In the example sentence this is indicated by a ① above the word;

- **intensity factor**

  Words that modify the intensity of the trend, from a small intensity (word "minor") to a higher intensity (word "major"). The selection of possible words is based on the amplitude of the gradient in the sector analysed. is In the example sentence this is indicated by a ② above the word;

- **features**

  Names of the analysed features, listed in the order of the analysis. The example sentence shows features X and Z ;

- **ranges**

  Interval where this analysis applies to. It is determined by the boundaries of each sector of the problem. The example sentence shows two ranges, one for each analysed feature: a_1 to a_n and b_1 to b_n.

Due to the limited number of options within a section (probability can increase, decrease or stay the same), the NLG technique we selected was one of dynamic templates (GATT; KRAHMER, 2018; REITER; DALE, 2000) as it is itself easily interpretable and manipulable.

The main procedure is as follows:

- For each class label:
  - Present the overall trend of the space analysed.
  - For each intensity captured by the analysis:
    - Describe the ranges of input features for the sector with this trend.

This procedure is replicated to describe each label within the boundaries determined by the restrictions of the sensibility value. In other words, labels that are not very relevant (according to the selected sensibility value) do not lead to any text. Also, this variable controls the size of each sector: the more sensitive an analysis is, the smaller the sectors are.

As the templates use the same sequence of words to describe each label, the previous procedure does not produce a fluid text. To circumvent this difficulty, some words in the output sentences were parameterized (Figure 3.8). That is, some words are replaced by a synonym (or by the combination of other words) during operation. It is also possible in our system to differentiate these markers from the ones used to fill in the values when sentences are generated. To control the synonyms used in the templates, a dictionary was implemented, as depicted by Figure 3.9. Each positional mark is replaced by a word or sentence, depending of the previous ones used before. The rationale behind this was to avoid using the same words on sequential sentences and, more importantly, to reduce the chance of a frequently used word to be used again.

```
elif type=="overall_text":
    return "\n[1] {0} have a {1} {2} [2] \
        features {3} increases.\n".format(*values_list)
```

Figure 3.8: Example of parameterized template (source: author (2020)).

```
self.synonyms_dict = {
            "[1]":[
                "Overall, class",
                "Generally, class",
                "Class",
                "It is possible to observe that class"],
            "[2]":[
                "in chance to occur when",
                "on the probability to be true when",
                "in the odds to occur when",
                "in expectation to be true when"
            ],
```

Figure 3.9: Fragment of the dictionary of synonyms (source: author (2020)).

The implementation of this method was loosely inspired in the term frequency-inverse document frequency (TF-IDF) metric, however the proposed metric tries to leave the most used words with less probability of reuse, instead of leaving them ordered from the most to least important. While the TF-IDF algorithm calculates the term frequency of tokens in a document and multiplies it by the inverse document frequency over all of the documents to find a balanced score of importance, the developed procedure sets a prior balanced sentence weight

(to start with an equal sentence importance) and dynamically redoes the calculations of importance. With each new sentence selected, its weight value is equally divided amongst the other sentences, and then the currently selected sentence weight is set to zero.

When a textual gap is to be filled, a set of synonyms are selected from the synonym dictionary. At the beginning, each one of these synonyms is selected with the same probability given by an option weight:

$$option\ weight = \frac{1}{total\ number\ of\ options}. \tag{3.2}$$

After a word is selected, the probability that it is selected again is recalculated so as to balance balance its frequency on the final text. Starting from the initial state (equal probability of selection), the following steps describe how this probability changes for each selection:

1. Apply a weighted random selection in the list of options, obtaining one option;
2. Distribute the current probability of usage of this options equally between the other options in this list;
3. Set the current probability of this option to 0.

The weight update steps can be expressed as follows:

$$\begin{aligned} w_j^{t+1} &= w_j^t + \frac{w_i}{N-1}, \forall w_j \neq i, \\ w_i^{t+1} &= 0, \end{aligned} \tag{3.3}$$

where
w_i^t: Weight of the randomly selected option;
w_i^{t+1}: New weight of the randomly selected option;
w_j^t: Weight of the other options;
w_j^{t+1}: New weight of the other options;
N: Number of options.

This algorithm generates a more fluid text, as there is no chance that close sentences contain the same words.

3.3 Implementation and discussion

The implementation of our approach was done using the Python language and several packages to manipulate data. The framework was made into a package named Natural Language

Explanations (NaLax) that can be easily imported into any code and applied to any classifier model. This section presents the resulting implementation and also describes what was learned during its process of development and the challenges identified during the implementation.

3.3.1 The NaLax package

With the usability of the framework in mind, the development focused on simple functions that can easily generate results. An explanation can be generated in a single command line:

```
explanation = ge.gen_global_explanation(X_train, features=feats)
```

The object **ge** is an instance of the **GenerateExplanations** class, which receives the model to be analysed, a list of names of features to be used in the PD function and the input data to calculate it (as seen on Figure 3.10).

The method **generate_global_text** returns a string with the report for the model given in the parameters. It manipulates the data sent within the parameters with the method **explainer** and generates a text with the **generate_behavior_text** method. The sensibility parameter, that controls the depth of analysis of the framework, is given to this function to regulate the output.

```
def gen_global_explanation(self, X, features = [],
                           sensibility_per_dimension=[0.35,0.45],
                           grid_resolution = 50, output_type = "text" ):

    nalax_explainer = nalax.explainer(self.model,features, X)

    return nalax_explainer.generate_behavior_text(
        sensibility_per_dimension = sensibility_per_dimension,
        grid_resolution = grid_resolution,
        output_type = output_type)
```

Figure 3.10: Generation of explanations on NaLaX code (source: author (2020)).

Figure 3.11 shows the outputs of a test code written to examine the simplicity of the package. In it, a menu was used to access the functions of fitting a model and then apply the technique in it. It is also possible to output the results to a text file, as NaLax can generate a quite long report if so desired.

An option was implemented so as to generate an explanation for a given individual sample. Figure 3.12 shows the output for a random sample of data extracted from the Wine Dataset.

This option was introduced after the comments received during the qualifier exam, where it was pointed out that several requests for explanations with respect to bank decisions are due to an interest in understanding a single decision. This solution follows the rationale described

```
1 - Fit wine quality Random Forest Model
2 - Generate global explanations
4 - Exit

Choose one option: 1
Model fitted.

1 - Fit wine quality Random Forest Model
2 - Generate global explanations
4 - Exit

Choose one option: 2
Enter the features to be analysed separated by commas:
alcohol,volatile acidity
Nalax output:

The analysed model predicts 8 different classes, and for each one of them
```

Figure 3.11: Using the NaLaX system (source: author (2020)).

in Section 3.1.4, where local explanation is achieved placing the desired sample point on the calculated surface of analysis.

Identifying the placement of the sample point, one can describe the trend in its sector so as to explain the decision. This explanation, however, may not match exactly the value in the input, as the algorithm calculates mean values over a sector. The higher the value of the sensibility variable, the more accurate this local analysis tend to be, since the sector in which the sample is placed is smaller.

3.3.2 Difficulty: The curse of dimensionality

Because PD functions must calculate predictions over all possible combinations of values for all of the features in a model, our proposal is heavily affected by the size and number of features of the dataset. Hence, NaLax takes a great deal of computational effort. During the work the existing libraries were examined to check whether they could be optimized, but it seems they are already quite efficient.

```
1 - Fit wine quality Random Forest Model
2 - Generate global explanations
3 - Generate local explanations
4 - Exit

Choose one option: 3
Given the current analysis, this model have an overall **minor** _increase
_ of chance to predict class 5, which was the one for this sample.
Specifically, the sampled data falls within the range 0.05333333333333333
and 0.12 for feature volatile acidity and 0.05333333333333333 and 0.12 for
 feature volatile acidity, which is a sector in which the model tend have
a **considerable** _increase_ of prediction for this class.
```

Figure 3.12: Local explanation (source: author (2020)).

Still, as an example of difficulties that may emerge, we can imagine a model with 10 features. If we wish to apply a PD function to only 2 of them, first we need to determine the ranges of values we want them to be analysed. Because each feature vary within different ranges, and the weight of the effects of each one in the model may be different, it is possible to determine how many points we wish to analyse. If we take 10 equally spaced points for each of the features we would have 100 combinations of points of the interested features. To create a surface of the PD function it is also needed to determine the sample points of the remainder features, which in this case sum 8 in total. In this scenario there are 10^8 combinations of values for these features, assuming the same 10 point for each one. Ultimately, each of the 100 points of the studied features is combined with these. All of the 10^8 points where the model would be applied in combination with the pair in study would have to be averaged, to finally obtain 1 point of the PD function surface.

4 EVALUATION

To evaluate our approach, we must test it with human subjects. This section discusses metrics that evaluate automatically generated texts and describes evaluation runs with human subjects.

The evaluation runs happened at two different stages of the project, and with two sources of data. The first dataset is the **Wine Quality Dataset** with widely known data used to predict the class of wines. The second dataset focuses on the worldwide pandemic of 2020. Several datasets have been published aiming to widen the access to data gathered by health professionals with respect to the COVID-19 virus. We used the **Diagnosis of COVID-19 and its Clinical Spectrum Dataset** provided by the Einstein Data4u project, from one of the most prestigious hospitals in Brazil. Models generated from this dataset can shed light on the pandemic, and as our framework generates complete textual information about the model, the information contained in it may help any person without further technical expertise and it can be used as documentation for future reference.

4.1 Wine Quality Dataset

The evaluation of our proposed approach was based on the generation of an explanation for the behavior of a random forest model with 300 trees, each with a maximum depth of 50. We focused on random forests because we wanted to emphasize that our approach is not solely geared towards large deep neural networks (for which model-specific methods exist). Other common classifiers can also be quite difficult to understand.

As noted previously, the dataset we used was the Wine Quality Dataset (CORTEZ, 2009), a dataset containing physicochemical properties related to red and white variants of a Portuguese wine, with a label determining the class of each sample, from 0 (very bad) to 10 (excellent).

4.1.1 Unit test

An example of the automatically generated text is presented in Figure 4.1, which describes how the probability of class 3 wines (the most relevant one) changes when features **alcohol** and **volatile acidity** vary within their range in the training dataset. The text comments on the probability of each class of wine given the training dataset.

Class 3 have a **considerable** decrease in chance to occur when features alcohol and volatile acidity increases.
Next, it is detailed 4 ranges of values so that it is possible to verify the output variation given the features values:
There is a **major** decrease of chance for this class to occur when:
- feature alcohol varies from value 0.14 to 0.46 and
- feature volatile acidity varies from value 0.05 to 0.18,
- feature alcohol varies from value 0.46 to 0.68 and
- feature volatile acidity varies from value 0.05 to 0.18,
- feature alcohol varies from value 0.46 to 0.68 and
- feature volatile acidity varies from value 0.18 to 0.39.

There is a **major** increase of chance for this class to occur when:
- feature alcohol varies from value 0.14 to 0.46 and
- feature volatile acidity varies from value 0.18 to 0.39.

Figure 4.1: Text result for the Wine Quality Data Set (source: author (2020)).

Note that this approach is not focused on showing whether the training is balanced, nor whether the model is properly encoding the data. The output conveys the behavior of the model in the space of possible values of the features.

Even though the sensitivity variable was set to 0.4 on average (meaning that the first sector will have 60% of the total size of the feature space) the generated text is quite long if we consider all labels (Figure 4.1 only contains one class out of eleven). The larger the sensitivity the smaller the sectors, causing the text to be more detailed.

It is to be expected that with a more complex problem (with more features and labels) the textual explanation for the model's behavior will be significantly longer. A longer report is not a problem: the granularity of the report can be controlled, so the end user can choose anything from a few paragraphs to a book-length description.

We can see that our system can generate a comprehensive text of the behavior of a model. The implementation of the sensitivity variable was the most important factor in text generation as it enabled granular control of the report.

4.1.2 Evaluation metrics

One metric that may come to mind when dealing with automated generated text is the BLEU score (PAPINENI, 2002). This scores rates a text on how close it is to a set of source sentences (generated by humans for a given context). Unfortunately, this metric is aimed at the evaluation of translations; in the present context there is no human generated text to compare to the output. Several other metrics that quantify the "complexity" of a text were examined, however none of them seemed to capture the information needed to classify the generated text as "good" or "bad". The literature that focuses on text evaluation tries to match the complexity level of a text grading it accordingly to the school level of a student or the maturity of a reader. This matching is achieved mainly by analysing the followings variables:

- Word frequency;
- Word length;
- Sentence length;
- Paragraph length.

Metrics and systems like the Lexile Framework (STENNER, 1996), Pearson Reading Maturity Metric (LANDAUER, 2011) (based on the LSA algorithm (LANDAUER; DUMAIS, 1997)), Coh-Metrix (GRAESSER, 2004), and many more well established metrics (Dale-Chall (DALE; CHALL, 1948), Flesch–Kincaid (KINCAID, 1975), Gunning Fog (GUNNING, 1968) and SMOG (MCLAUGHLIN, 1969)) have been constructed based on these variables, and others derived from them, to rate a text with respect to its complexity or level of difficulty (readability).

The SMOG metric, used to measure the readability of text, which estimates the years of education needed to understand it, is calculated as follows (the other metrics have similar approaches):

$$grade = 1.043 * \sqrt{number\ of\ polysyllables * \frac{30}{number\ of\ sentences}} + 3.291, \qquad (4.1)$$

where the variable **number of sentences** must be at least 30 in the analysed corpus and the variable **number of polysyllables** are words in these sentences that have more than 3 syllables.

Because the generated text here is based on templates, any metrics derived from this variables would be static or would not vary much so none be used to evaluate our results. This

difficulty led to the development of a measurement scheme that could adequately evaluate our approach. Section 4.1.3 describes the metrics that enable fair comparison between the established state-of-the-art techniques and the one proposed here, and that try to follow the notion of a good explanation indicated by Miller (2019).

4.1.3 Experiment with users

To evaluate the effective application of our approach on human readers, an experiment was conducted on a machine learning class of 48 students from a Master Engineering degree. These students heard first a 30 minute explanation about interpretability and several tools related to the field, focusing on LIME, PDP and the proposed framework, then they were asked to participate on the evaluation experiment.

This experiment consisted of showing the results obtained by two state of the art interpretability model-agnostic techniques: PDP and LIME, an then showing the results obtained by our approach so as to compare techniques. For each technique presented, students were asked to score with respect to three criteria, as shown on Figure 4.2.

1. **Quick understanding of information:** How much effort the student spent to understand the underlying information represented by the technique.

2. **Seemingly reliable result:** Whether the result given by the technique appeared correct and reliable.

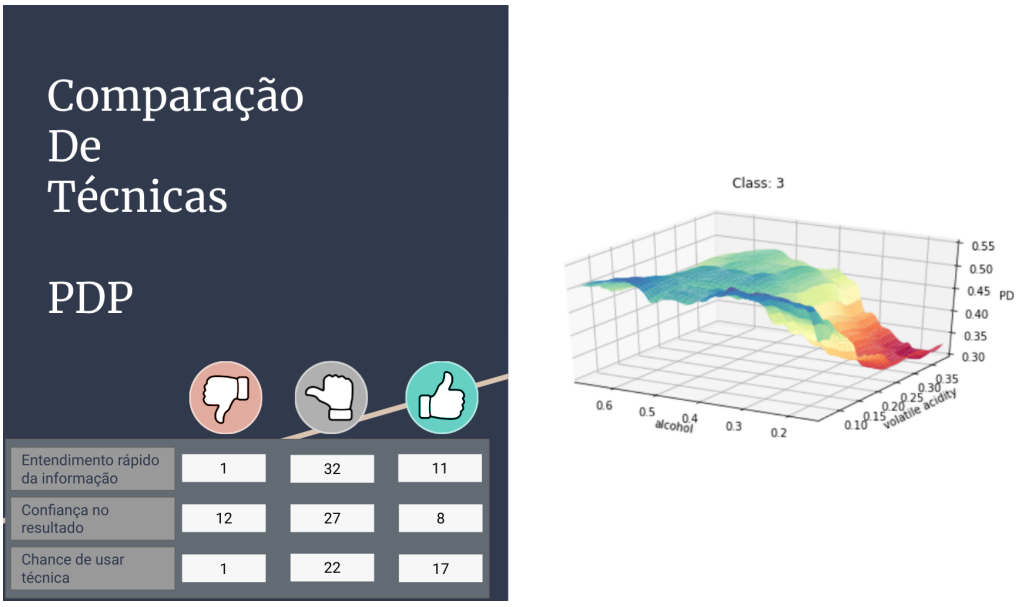


Figure 4.2: Example of screen presented to students (source: author (2020)).

Table 4.1: Results of the experiment

Technique	Quick understanding of information			Seemingly reliable result			Chance of usage		
	Negative	Neutral	Positive	Negative	Neutral	Positive	Negative	Neutral	Positive
PDP	1	36	11	12	28	8	1	30	17
LIME	2	29	17	2	22	24	2	22	24
Proposed Technique	3	16	29	5	30	13	4	26	18

3. **Chance of usage:** Whether the student would use the technique in a real project.

These criteria were related to "satisfaction", each one of them with possible values: **Positive**, **Neutral** and **Negative**. For instance, a positive satisfaction for the first criterion means that little effort was spent to understand the information provided by the method.

The evaluation results presented in Table 4.1 indicate that our approach outperformed other approaches with respect to the first criterion, while it was equally as satisfactory with respect to the other criteria — suggesting that textual explanations led to a more satisfactory understanding. This is particularly interesting in our setting as students were knowledgeable about mathematical expressions and graphs; even then text was the preferred.

It is important to point out that this experiment was performed within the ethical requirements required by the ethics council in the university – *Comitê de Ética em Pesquisa do Hospital Universitário da USP* (CEP-HU/USP) – to conduct researches with individuals. After presenting the methodology of this experiment, a representative of this council considered it to be a public opinion poll, therefore no submission to a special analysis was needed. The response given by the Committee Secretary can be found in the Appendix B.

4.2 Diagnosis of COVID-19 and its Clinical Spectrum Dataset

Given the dire situation caused by the COVID-19 pandemic, we have examined the benefits of interpretability techniques over models trained with related medical data. In such cases, where the interpretation of the results is as important as the result itself, our proposal can be very useful for professionals to extract as much information as possible from fitted models. For instance, a medical technician may evaluate whether a model is coherent with medical knowledge in the literature.

The Diagnosis of COVID-19 and its Clinical Spectrum Dataset was made available to the public at the Kaggle platform by the researchers of the Einstein Institute. In this dataset there are over 5.500 entries of patients labeled with positive and negative to COVID-19. The proportion of positive cases in the dataset is 0.099; over 80% of the data related to blood samples are

missing. Hence the analysis is complex and demands imputation strategies and techniques to deal with the unbalanced labels.

4.2.1 Unit test

By processing the COVID-19 dataset we produced a Gradient Boost Classifier with 100 estimators, with a maximum depth of 5, using 35 out of the 110 columns available in the dataset. The reduced number of columns was due the removal of correlations and mostly null columns. At the same time, we also developed an interface to make the framework more portable and accessible to anyone interested in using it to improve interpretability of any classifier.

The result of this development is depicted in Figure 4.3. In this interface the user can run the model with the desired features and see the resulting explanatory text after the process is finished.

The generated text for the Gradient Boost Classifier is depicted in Figure 4.4. The text describes how the probability that class 1 (positive to COVID-19) holds changes when the

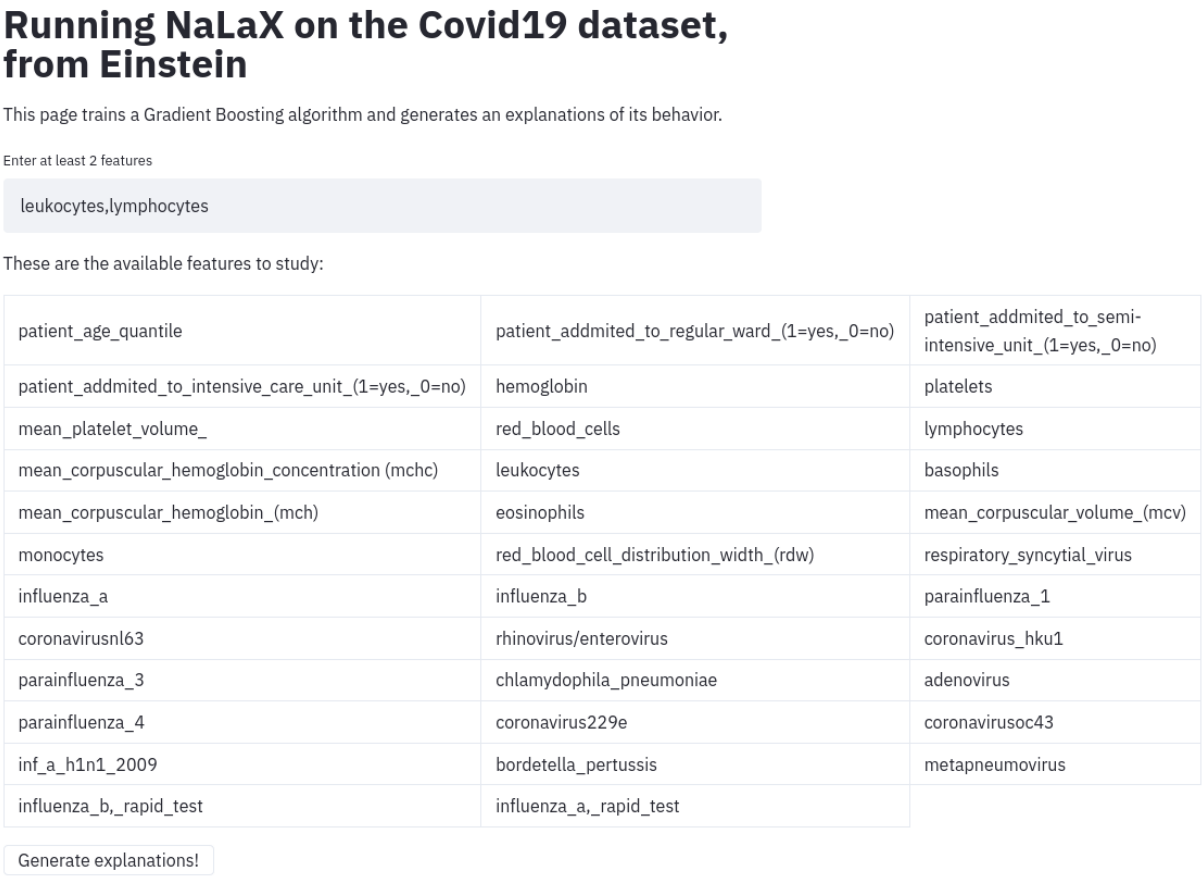


Figure 4.3: Web interface (source: author (2020)).

values of leukocytes and lymphocytes vary.

The analysed model predicts 1 class,and this text details how some features influence its chance of prediction.
Generally, class 1 have a **major** *increase* on the probability to be true when features leukocytes and lymphocytes increases.
For this same class, it is detailed above 16 ranges of values that exposes the effect of the features on the output:
There is a **major** *increase* of chance for this class to occur when:
- feature leukocytes varies from value -0.6344 to -0.5014 and
- feature lymphocytes varies from value -0.0592 to -0.0204,
- feature leukocytes varies from value -0.5014 to -0.3951 and
- feature lymphocytes varies from value -0.0592 to -0.0204,
- feature leukocytes varies from value -0.3951 to -0.2664 and
- feature lymphocytes varies from value -0.0592 to -0.0204,
- feature leukocytes varies from value -0.2664 to -0.2115 and
- feature lymphocytes varies from value -0.0592 to -0.0204.

There is a **minor** *increase* of chance for this class to occur when:
- feature leukocytes varies from value -0.3951 to -0.2664 and
- feature lymphocytes varies from value -0.2157 to -0.0592,
- feature leukocytes varies from value -0.3951 to -0.2664 and
- feature lymphocytes varies from value 0.0744 to 0.1627.

There is a **major** *increase* of chance for this class to occur when:
- feature leukocytes varies from value -0.6344 to -0.5014 and
- feature lymphocytes varies from value -0.2157 to -0.0592,
- feature leukocytes varies from value -0.6344 to -0.5014 and
- feature lymphocytes varies from value -0.0204 to 0.0744,
- feature leukocytes varies from value -0.6344 to -0.5014 and
- feature lymphocytes varies from value 0.0744 to 0.1627.

There is a **less significant** *increase* of chance for this class to occur when:
- feature leukocytes varies from value -0.5014 to -0.3951 and
- feature lymphocytes varies from value -0.2157 to -0.0592,
- feature leukocytes varies from value -0.2664 to -0.2115 and
- feature lymphocytes varies from value -0.2157 to -0.0592,
- feature leukocytes varies from value -0.5014 to -0.3951 and
- feature lymphocytes varies from value -0.0204 to 0.0744,
- feature leukocytes varies from value -0.3951 to -0.2664 and
- feature lymphocytes varies from value -0.0204 to 0.0744,
- feature leukocytes varies from value -0.2664 to -0.2115 and
- feature lymphocytes varies from value -0.0204 to 0.0744,
- feature leukocytes varies from value -0.5014 to -0.3951 and
- feature lymphocytes varies from value 0.0744 to 0.1627,
- feature leukocytes varies from value -0.2664 to -0.2115 and
- feature lymphocytes varies from value 0.0744 to 0.1627.

Figure 4.4: Text result for the Diagnosis of COVID-19 and its Clinical Spectrum Dataset (source: author (2020)).

The long text output presented in Figure 4.4 was produce by a sensibility value set higher

value to what we did in Section 4.1.1. One can verify details about the classifier with higher granularity.

4.2.2 Experiment with users

A survey through Google Forms was developed aiming to validate the explanation given by our approach. The form was presented to a body of more proficient machine learn practitioners. A total of 49 people participated in the survey.

The survey contained three questions; one of these questions is shown in Figure 4.5 in connection with a PDP analysis. The participants were also asked to write a text regarding the techniques presented, where they could openly express their opinions about strengths and weaknesses of each one.

It is important to note that this survey was also in accordance with the directives given by CEP-HU/USP as the same protocol was employed and no personal and sensitive information was asked from the participants.

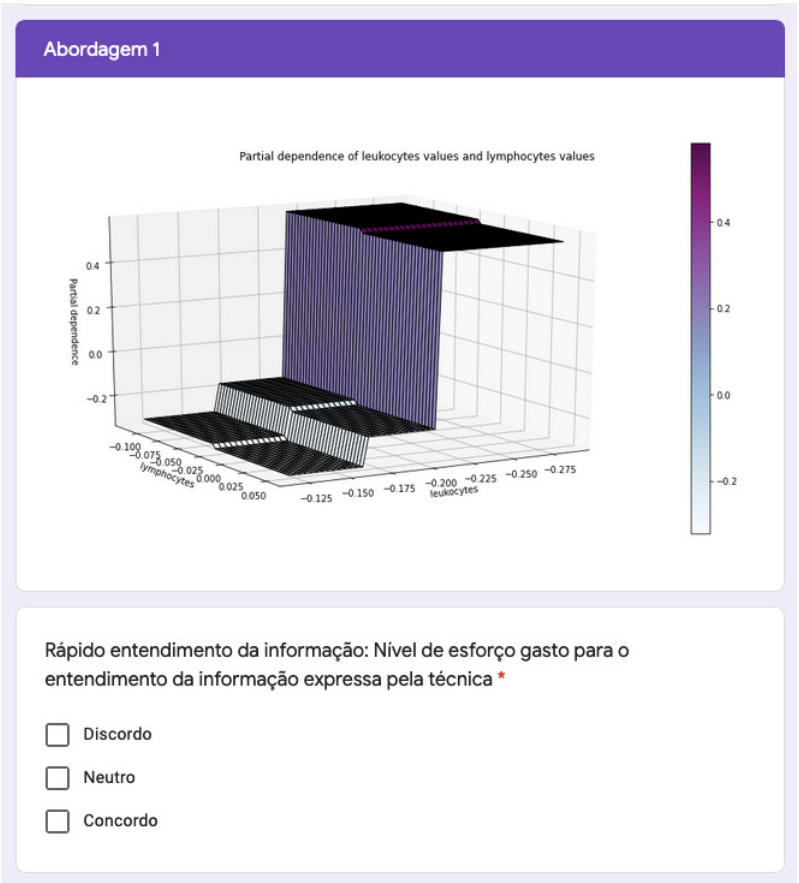


Figure 4.5: Question example in Google Forms (source: author (2020)).

Each participant first read a text on the importance of interpretability in machine learning.

Then the participant was presented an image with the output of each technique, and an automatic explanation about how the output should be interpreted.

Finally, participants rated techniques, again with respect to Section 4.1.3:

1. Quick understanding of information;
2. Seemingly reliable result;
3. Chance of usage.

The possible values were, again: Positive, Neutral and Negative. These values capture the subjective feeling regarding each technique.

In Figure 4.6 we see the effect of producing too much detail in NaLax when the user is not explicitly asking for it: other methods are better evaluated. When we examined the positive responses we can see that the perception of "easiness in reading" is indeed dependent on the individual.

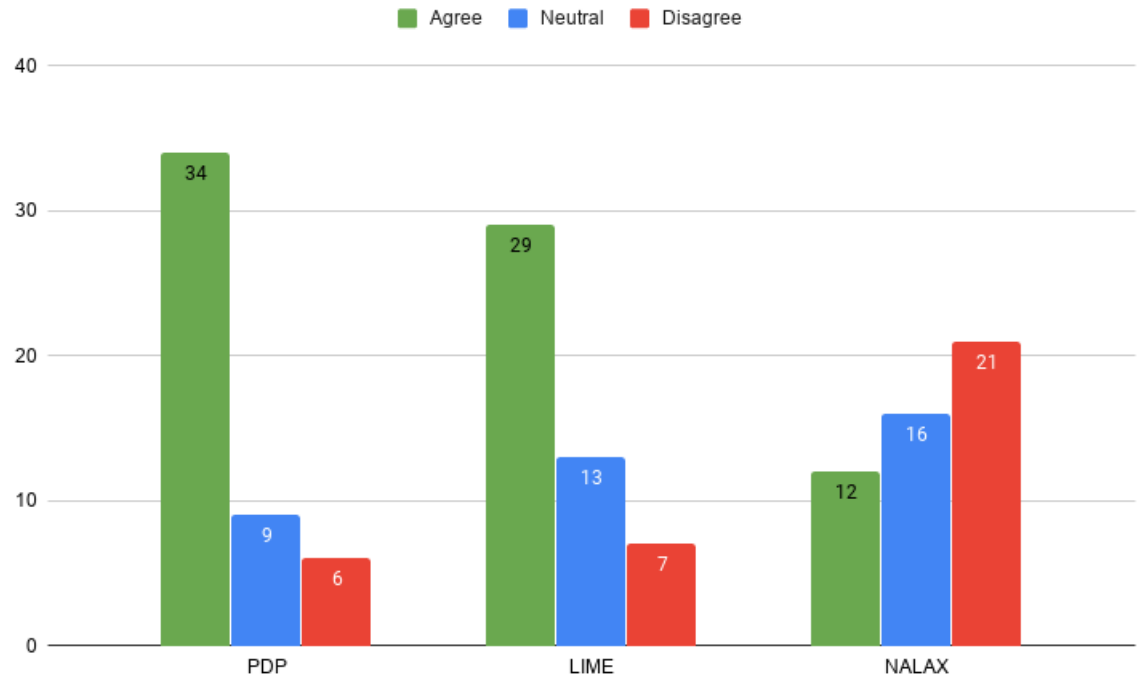


Figure 4.6: Quick understanding of information (source: author (2020)).

In contrast with the previous result, using a textual representation as the channel of communication seems to bring a higher level of "trustfulness" that surpasses charts. As seen on Figure 4.7, NaLax gets very positive results and surpasses PDP. In this evaluation, the proposed

approach received almost equivalent responses as LIME, indicating that it can be considered really comparable to the state-of-the art techniques of the literature.

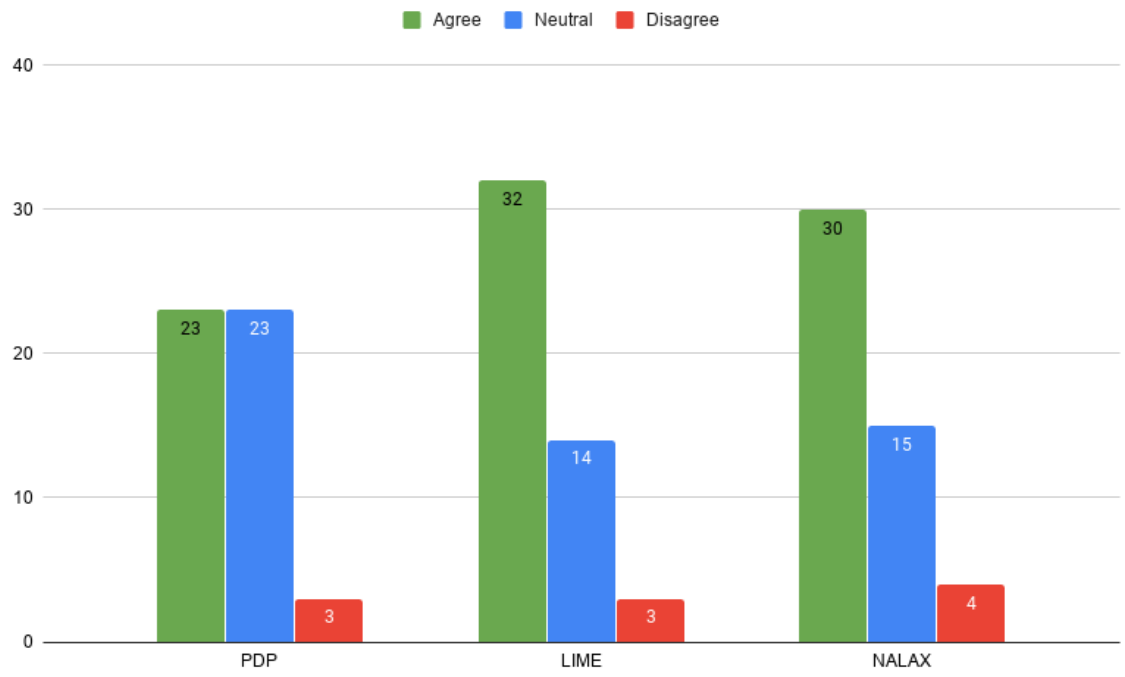


Figure 4.7: Seemingly reliable (source: author (2020)).

More modest results are presented at Figure 4.8, where NaLax had more "balanced" scores and was not as good as the others. However, in contrast with Figure 4.6 we can see that that even though the general perception was that the approach is not as fast in transferring the information compared to the others, participants still like it and consider it for use in the future.

Complementing the quantitative results, the analysis of the texts produced by the participants students corroborated several facts, and exposed possibilities which were not the main objective at the beginning of the research. Also, these comments enabled a deeper view of the users' perspectives regarding the approach, which could not be perceived by the survey.

Overall, there were mixed comments regarding the level of detail of the text and the amount of information in it. Some students were not comfortable reading an extensive text, and felt that this was a negative aspect of the approach, while other students pointed out that the lengthy texts were useful as they thoroughly explained the effects of the model. More importantly, the granularity control was deemed positive as one could choose how extensive this resulting text could be.

As the generation of explanation – even if extensive – was "by design", some of these comments reflect the expected portion of users which would not benefit of this level of detail.

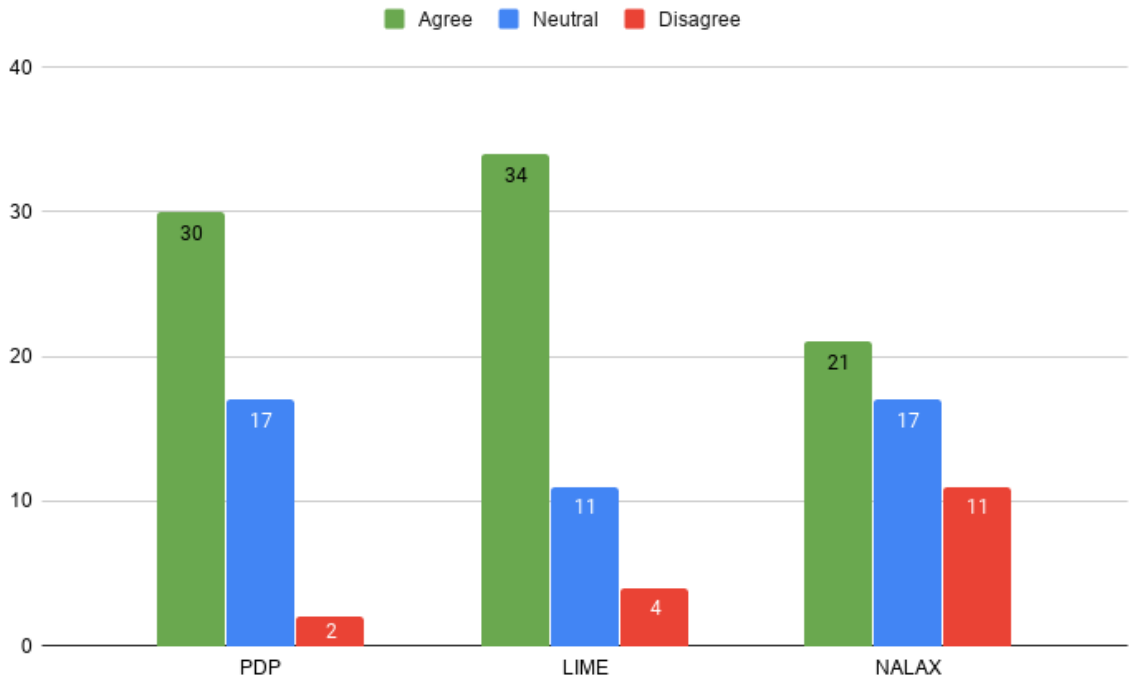


Figure 4.8: Interest in future use (source: author (2020)).

Because the NaLaX implementation is able to regulate the length of its results using a sensibility variable, on a real scenario these users could set the value of it to the level of their need and have a more concise explanation (as the one on Section 4.1.3).

The comments presented at Appendix C are excerpts of responses given by a sample of students that found our approach to be positive, and present their reason over this opinion. One particular comment emphasized the fact that NaLax is useful to transfer information as it uses natural language, avoiding abstractions that could create a barrier to understanding and sometimes needs to be prior learned. It is especially important to have a textual representation as there are situations where one needs to present information to a visually impaired reader, and it is not trivial to translate a visual representation of data into information when in this situation.

These results were helpful to expand the opportunities of our approach and brought to light some aspects that could be better explored. Even the “negative” comments were useful to be aware of real use scenarios where extensive texts can be harmful to the user understanding.

5 CONCLUSION AND FUTURE WORK

This document described the generation of textual explanations that clarify the behavior of classifiers. State of the art model-agnostic interpretability techniques were used and new algorithms were developed to analyze a classifier through a series of dynamic template-based sentences. In the proposed method, explanations can be generated with the level of detail determined by the user. The resulting framework generates global explanations that help the user identify the effect of features. We have also developed a procedure that adapts the global analysis into a local analysis explaining the behavior of a given sample. This sort of local explanation is, of course, dependent on the level of detail selected by the user.

A few evaluation experiments were designed and applied to evaluate the quality and usability of the proposed explanation techniques. The tests captured the quality of the analysis and the usability of the procedures with real users. The first test, run in 2019 with a partial implementation of the framework, had undergraduate students to compare the present proposal against two state-of-the-art techniques, all applied to a well-known dataset in the literature. This test produced positive responses regarding the proposed method. The same testing procedure was again applied in 2020 with more proficient researchers (master degree candidates) who took part in the survey. This second evaluation also produced positive results, placing the proposed approach amongst the state-of-the-art techniques in the literature.

Future development of this work should improve text generation with regard to finely detailed sections (which are now quite repetitive). Improvements can be achieved by implementing new textual representations for value intervals, combined with the weighted selection algorithm described in Section 3.2. Also, the development of heuristics that can reduce the exponential complexity problem of our proposal would expand its applications to higher dimensional settings. Another possible way to continue this would would be to develop a chatbot interface that could present explanations interactively. One could then study every facet of a model without a single line of code, so as to get a more adequate response than a parameterized textual template.

REFERENCES

AQUINO, R. M. D.; COZMAN, F. Natural Language Explanations of Classifier Behavior. In: *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. [S.l.: s.n.], 2019. p. 239–242.

BACH, S.; BINDER, A.; MONTAVON, G.; KLAUSCHEN, F.; MÜLLER, K.-R.; SAMEK, W. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLOS ONE*, Public Library of Science, v. 10, n. 7, p. e0130140, jul 2015. ISSN 1932-6203.

BIRAN, O.; MCKEOWN, K. Human-Centric Justification of Machine Learning Predictions. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. [S.l.]: AAAI Press, 2017. (IJCAI'17), p. 1461–1467. ISBN 978-0-9992411-0-3.

CONDE-CLEMENTE, P.; ALONSO, J. M.; TRIVINO, G. rLDCP: R Package for Text Generation From Data. In: *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. [S.l.]: IEEE, 2017. p. 1–6. ISBN 978-1-5090-6034-4.

CORTEZ, P.; CERDEIRA, A.; ALMEIDA, F.; MATOS, T.; REIS, J. Modeling Wine Preferences by Data Mining From Physicochemical Properties. *Decision Support Systems*, North-Holland, v. 47, n. 4, p. 547–553, nov 2009. ISSN 0167-9236.

DALE, E.; CHALL, J. S. *A Formula for Predicting Readability: Instructions*. [S.l.]: Taylor & Francis, Ltd., 1948. 37–54 p.

DARPA-BAA-16-53. Explainable Artificial Intelligence (XAI). *Defense Advanced Research Projects Agency*, p. 1, 2016.

DOS SANTOS, C. N.; GATTI, M. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In: *COLING 2014 - 25th International Conference on Computational Linguistics, Proceedings of COLING 2014: Technical Papers*. [S.l.: s.n.], 2014. ISBN 9781941643266.

DOSHI-VELEZ, F.; KIM, B. Towards A Rigorous Science of Interpretable Machine Learning. *arXiv:1702.08608*, 2017.

European Union. Regulation 2016/679 of the European Parliament and the Council of the European Union. *Official Journal of the European Communities*, p. 1–88, 2016.

FORNBERG, B. Generation of Finite Difference Formulas on Arbitrarily Spaced Grids. *Mathematics of Computation*, v. 51, n. 184, p. 699–699, 1988. ISSN 0025-5718.

GATT, A.; KRAHMER, E. Survey of the State of the Art in Natural Language Generation: Core Tasks, Applications and Evaluation. *Journal of Artificial Intelligence Research*, v. 61, p. 1–64, 2018. ISSN 10769757.

GOLDSTEIN, A.; KAPELNER, A.; BLEICH, J.; PITKIN, E. Peeking Inside the Black Box: Visualizing Statistical Learning With Plots of Individual Conditional Expectation. *Journal of Computational and Graphical Statistics*, v. 24, n. 1, p. 44–65, sep 2015. ISSN 15372715.

GRAESSER, A. C.; MCNAMARA, D. S.; LOUWERSE, M. M.; CAI, Z. Coh-Metrix: Analysis of Text on Cohesion and Language. *Behavior Research Methods, Instruments, & Computers*, Springer-Verlag, v. 36, n. 2, p. 193–202, may 2004. ISSN 0743-3808.

GUIDOTTI, R.; MONREALE, A.; RUGGIERI, S.; TURINI, F.; GIANNOTTI, F.; PEDRESCHI, D. A Survey of Methods for Explaining Black Box Models. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 51, n. 5, p. 93:1–93:42, ago. 2018. ISSN 0360-0300.

GUNNING, R. *The Technique of Clear Writing*. [S.l.]: McGraw-Hill, 1968. 329 p. ISBN 9780070252066.

HAGRAS, H. Toward Human-Understandable, Explainable AI. *Computer*, v. 51, n. 9, p. 28–36, sep 2018. ISSN 15580814.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning*. New York, NY: Springer New York, 2009. (Springer Series in Statistics). ISBN 978-0-387-84857-0.

HECHTLINGER, Y. Interpretation of Prediction Models Using the Input Gradient. *arXiv:1611.07634*, 2016.

HOLZINGER, A.; BIEMANN, C.; PATTICHIS, C. S.; KELL, D. B. What Do We Need to Build Explainable AI Systems for The Medical Domain? dec 2017.

KELLEY, R.; BARRY, R. Sparse spatial autoregressions. *Statistics Probability Letters*, v. 33, n. 3, p. 291 – 297, 1997.

KINCAID, J. P.; FISHBURNE JR, R. P.; ROGERS, R. L.; CHISSOM, B. S. *Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel*. 1975.

LANDAUER, T. K.; DUMAIS, S. T. A Solution To Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*, US: American Psychological Association, v. 104, n. 2, p. 211–240, 1997. ISSN 1939-1471.

LANDAUER, T. K.; KIREYEV, K.; PANACCIONE, C. Word Maturity: A New Metric for Word knowledge. *Scientific Studies of Reading*, v. 15, n. 1, p. 92–108, jan 2011. ISSN 10888438.

MCLAUGHLIN, G. H. SMOG Grading - A New Readability Formula. *Journal of Reading*, [Wiley, International Reading Association], v. 12, n. 8, p. 639–646, 1969.

MILLER, T. Explanation in Artificial Intelligence: Insights From The Social Sciences. Elsevier, v. 267, p. 1–38, feb 2019. ISSN 00043702.

MONTAVON, G.; LAPUSCHKIN, S.; BINDER, A.; SAMEK, W.; MÜLLER, K.-R. Explaining Nonlinear Classification Decisions With Deep Taylor Decomposition. *Pattern Recognition*, v. 65, p. 211–222, may 2017. ISSN 00313203.

NAZARI, N.; MAHDAVI, M. A Survey on Automatic Text Summarization. *Journal of AI and Data Mining*, v. 0, n. 0, 2018. ISSN 2322-5211.

NOVAK, V.; PERFILIEVA, I.; MOCKOR, J. *Mathematical Principles of Fuzzy Logic*. [S.l.]: Springer US, 1999. 320 p. ISBN 9781461552178.

PAPINENI, K.; ROUKOS, S.; WARD, T.; ZHU, W.-J. BLEU: A Method for Automatic Evaluation of Machine Translation. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002. (ACL '02), p. 311–318.

PURI, N.; GUPTA, P.; AGARWAL, P.; VERMA, S.; KRISHNAMURTHY, B. MAGIX: Model Agnostic Globally Interpretable Explanations. *arXiv:1706.07160*, 2017.

REITER, E.; DALE, R. Building Applied Natural Language Generation Systems. *Natural Language Engineering*, 1997. ISSN 14698110.

____. *Building Natural Language Generation Systems*. [S.l.]: Cambridge University Press, 2000. 248 p. ISBN 0521620368.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. "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*. New York, NY, USA: ACM, 2016. (KDD '16), p. 1135–1144. ISBN 978-1-4503-4232-2.

STENNER, A. J. *Measuring Reading Comprehension with the Lexile Framework*. [S.l.], 1996.

VLEK, C. S.; PRAKKEN, H.; RENOOIJ, S.; VERHEIJ, B. A Method for Explaining Bayesian Networks for Legal Evidence With Scenarios. *Artificial Intelligence and Law*, Springer Netherlands, v. 24, n. 3, p. 285–324, sep 2016. ISSN 0924-8463.

WOŁK, K.; MARASEK, K. Neural-based Machine Translation for Medical Text Domain. Based on European Medicines Agency Leaflet Texts. In: *Procedia Computer Science*. [S.l.: s.n.], 2015. ISSN 18770509.

APPENDIX A – ALGORITHM

The algorithm in this appendix determines the sectors in the PD function surface and calculates the mean gradient for each one of these sectors. Each label is treated separately, as limited by a sensitivity vale that bounds the effect of labels and controls the size of the sectors.

Comments in the code explain each one of the steps.

Algorithm 1 Gradient Analysis

Input: *pd_data* (PD data for one particular class)
Output: list of $(bound\_index, mean\_gradient)$

```
### Initializing variables ###
slices_perc_sizes ← percentual size for each feature in PD
dim_stride ← empty list
num_features ← length(pd_data.shape)
### Loop to determine the stride in each dimension ###
for i = 0 to num_features − 1 do
    shape_size ← pd_data.shape[i]
    perc_size ← slices_perc_sizes[i]
    dim_stride.append(⌊shape_size ∗ perc_size⌋)
end for
do_loop ← True
slices ← empty list
indexes ← list with size num_features, filled with zeroes
### Loop to create the bounds of the sectors ###
while do_loop = True do
    bound_index ← empty list
    for i = 0 to num_features − 1 do
        last_index ← indexes[i] + dim_stride[i]
        if last_index ≥ pd_data.shape[i] then
            last_index ← pd_data.shape[i] − 1
        end if
        bound_ind ← bound_ind + (indexes[i], last_index)
    end for
    ### Calculates the mean value in this sector ###
    mean_grad = mean(grad(pdp_data[bound_ind]))
    slices.append((bound_ind, mean_grad))
    indexes[0] = indexes[0] + dim_stride[0]
    for i = 1 to num_features − 1 do
        if indexes[i − 1] ≥ pdp_data.shape[i − 1] then
            indexes[i] = indexes[i] + dim_stride[i]
            indexes[i − 1] = 0
        end if
    end for
    if indexes[−1] ≥ pdp_data.shape[−1] then
        do_loop = False
    end if
end while
return slices
```

APPENDIX B – CEP-USP MAIL

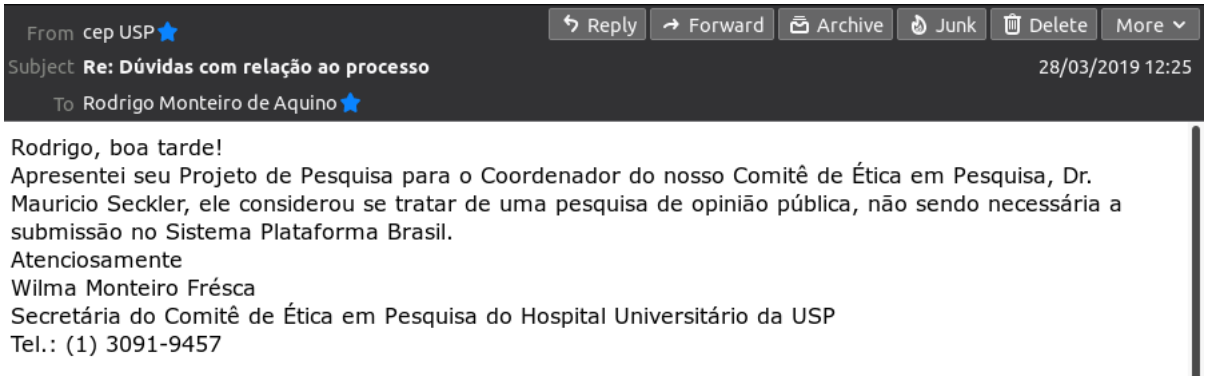
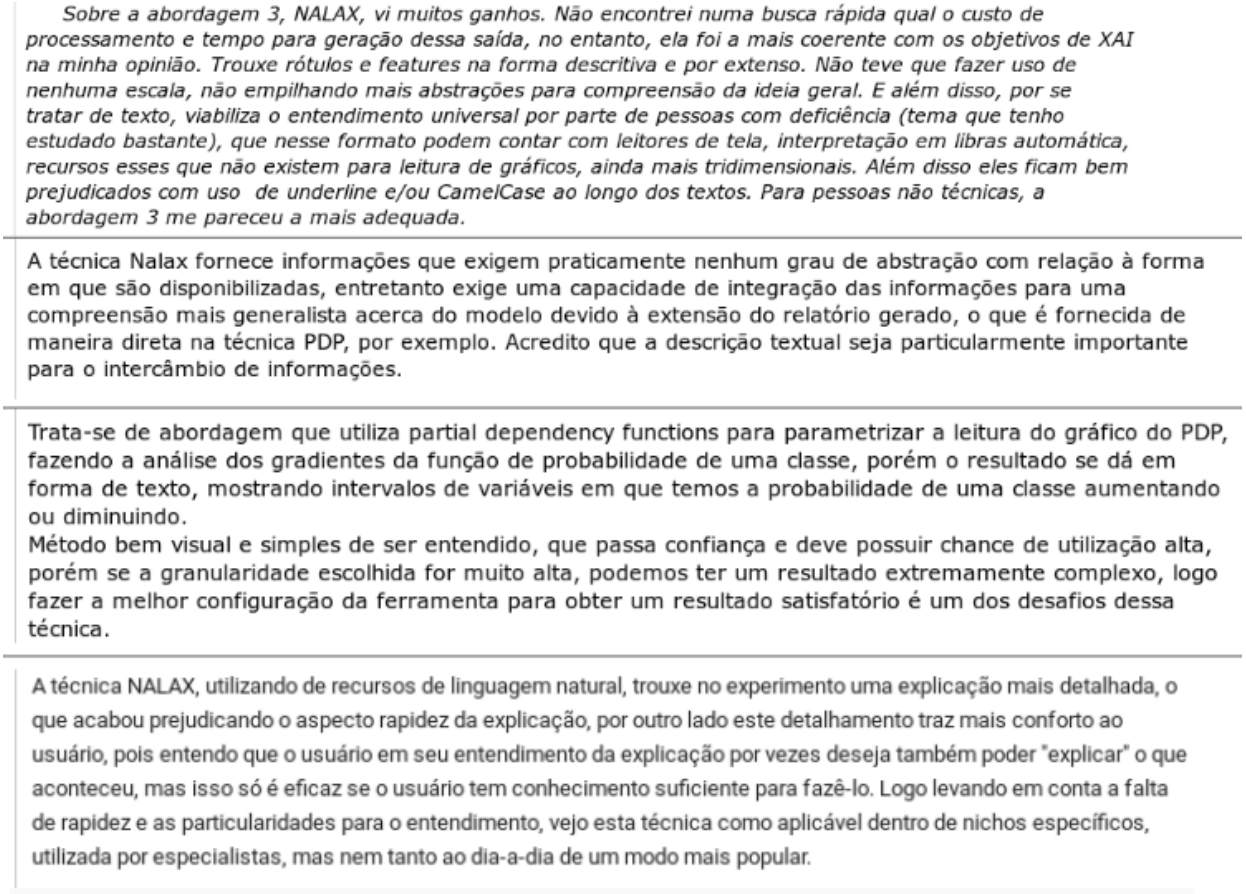


Figure B.1: CEP-USP message on ethical committee analysis (source: author (2019)).

APPENDIX C – COMMENTS REGARDING THE TECHNIQUE

Comments that highlights the relevance of the approach



Sobre a abordagem 3, NALAX, vi muitos ganhos. Não encontrei numa busca rápida qual o custo de processamento e tempo para geração dessa saída, no entanto, ela foi a mais coerente com os objetivos de XAI na minha opinião. Trouxe rótulos e features na forma descritiva e por extenso. Não teve que fazer uso de nenhuma escala, não empilhando mais abstrações para compreensão da ideia geral. E além disso, por se tratar de texto, viabiliza o entendimento universal por parte de pessoas com deficiência (tema que tenho estudado bastante), que nesse formato podem contar com leitores de tela, interpretação em libras automática, recursos esses que não existem para leitura de gráficos, ainda mais tridimensionais. Além disso eles ficam bem prejudicados com uso de underline e/ou CamelCase ao longo dos textos. Para pessoas não técnicas, a abordagem 3 me pareceu a mais adequada.

A técnica Nalax fornece informações que exigem praticamente nenhum grau de abstração com relação à forma em que são disponibilizadas, entretanto exige uma capacidade de integração das informações para uma compreensão mais generalista acerca do modelo devido à extensão do relatório gerado, o que é fornecida de maneira direta na técnica PDP, por exemplo. Acredito que a descrição textual seja particularmente importante para o intercâmbio de informações.

Trata-se de abordagem que utiliza partial dependency functions para parametrizar a leitura do gráfico do PDP, fazendo a análise dos gradientes da função de probabilidade de uma classe, porém o resultado se dá em forma de texto, mostrando intervalos de variáveis em que temos a probabilidade de uma classe aumentando ou diminuindo.
Método bem visual e simples de ser entendido, que passa confiança e deve possuir chance de utilização alta, porém se a granularidade escolhida for muito alta, podemos ter um resultado extremamente complexo, logo fazer a melhor configuração da ferramenta para obter um resultado satisfatório é um dos desafios dessa técnica.

A técnica NALAX, utilizando de recursos de linguagem natural, trouxe no experimento uma explicação mais detalhada, o que acabou prejudicando o aspecto rapidez da explicação, por outro lado este detalhamento traz mais conforto ao usuário, pois entendo que o usuário em seu entendimento da explicação por vezes deseja também poder "explicar" o que aconteceu, mas isso só é eficaz se o usuário tem conhecimento suficiente para fazê-lo. Logo levando em conta a falta de rapidez e as particularidades para o entendimento, vejo esta técnica como aplicável dentro de nichos específicos, utilizada por especialistas, mas nem tanto ao dia-a-dia de um modo mais popular.

Figure C.1: Students' comments regarding the technique (source: author (2020)).