Towards a Generic Framework for Black-box Explanations of Algorithmic Decision Systems

Clement Henin, Daniel Le Métayer

Towards a Generic Framework for Black-box Explanations of Algorithmic Decision Systems (extended version)

Clément Henin, Daniel Le Métayer

RESEARCH REPORT

N° 9276

Mai 2019

Project-Team PRIVATICS

ISSN 0249-6399   ISRN INRIA/RR--9276--FR+ENG

Towards a Generic Framework for Black-box Explanations of Algorithmic Decision Systems (extended version)

Clément Henin, Daniel Le Métayer

Project-Team PRIVATICS

Research Report n° 9276 — Mai 2019 — 24 pages

Abstract: The main goal of this paper is to define a generic framework for black-box explanation methods in order to make it easier to compare and classify different approaches. We focus on two components of this framework, called respectively "Sampling" and "Generation", which are characterized formally and used to build a taxonomy of explanation methods. This document gives details on how this framework can be used to describe black-box explanation methods found in the literature.

Key-words: Automatic decision system, explanability, algorithm transparency, black-box model, machine-learning, artificial intelligence

Vers un cadre générique pour l'explication des systèmes de décision algorithmique en mode boîte noire (version étendue)

Résumé : L'objectif principal de ce document est de définir un cadre génvrique pour les méthodes d'explication des boîtes noires afin de faciliter la comparaison et la classification des différentes approches. Nous nous concentrons sur deux composantes de ce cadre, appelées respectivement "Échantillonnage " et " Génération ", qui sont caractérisées formellement et utilisées pour construire une taxonomie des méthodes d'explication. Ce document détaille comment ce cadre peut être utilisé pour décrire les méthodes d'explication des boîtes noires que l'on trouve dans la littérature.

Mots-clés : Système de décision automatique, explabilité, transparence des algorithmes, modèle boite-boire, machine-learning, intelligence artificielle

Contents

1 Introduction

This document is an extended version to the conference paper ”Towards a Generic Framework for Black-box Explanations of Algorithmic Decision Systems” submitted to the Workshop Explainable AI of IJCAI 2019 appendices and a figure were added. In this paper, we defined a generic framework to describe black box explanation methods (BEMs) found in the literature. Because of space limitation in the original paper, we were not able to give a full description of how the framework instantiate for every method. This document in meant to do so.

2 Motivations

Algorithmic Decision Systems (hereafter “ADS”) are increasingly used in many areas, sometimes with a major impact on the lives of the people affected by the decisions. Some of these systems make automatic decisions, for example to reduce or to increase the speed of an autonomous car, while others only make suggestions that a human user is free to follow or not. In some cases, the user is a professional, for example a medical practitioner or a judge, while in other cases he is an individual, for example an internet user or a consumer. Some ADS rely on traditional algorithms, while others are based on machine learning (hereafter “ML”) and involve representations such as neural networks, Bayesian networks or decision trees. Regardless of these considerations, when an ADS can have a significant impact its design and validation should ensure a high level of confidence that it will meet its requirements.

Because the most accurate ML techniques often produce opaque ADS and opacity is a source of mistrust, explainability has generated increased interest during the last decade. Indeed, even if explanations are not necessarily a panacea, they can be very useful, not only to enhance trust in the system, but also to allow its users to understand its outputs and make proper use of it. Explanations can take different forms, they can target different types of users (hereafter “explainees”) and different types of techniques can be used to produce them. In this paper, we focus on techniques, called ”black-box”, that do not make any assumption of the availability of the code of the ADS or its implementation techniques. The only assumption is that input data can be provided to the model M and its output data can be observed.

Explainability is a fast growing research area and many papers have been published on this topic during the last years. These papers define methods to produce different types of explanations in different ways but they also share a number of features. The main goal of this paper is to bring to light a common structure for Black-box Explanation Methods (BEM) and define a generic framework in order to make it easier to compare and classify different approaches. This framework consists of three components, called respectively “Sampling”, “Generation” and “Interaction”. The need to conceive an explanation as a conversation rather than a static object, which is captured by our “Interaction” component, has been forcefully argued by several authors [17]. It must be acknowledged, however, that most contributions in the XAI community do not emphasize this aspect. Therefore, in view of space limitation, we focus on the “Sampling” and “Generation” components in this paper. We characterize these components formally and use them to build a taxonomy of explanation methods. We come back to the link with the “Interaction” component in the conclusion. Beyond its interest as a systematic presentation of the state of the art, we believe that this framework can also provide new insights for the design of new explanation systems. For example, it may suggest new combinations of Sampling and Generation components or criteria to choose the most appropriate combination to produce a given type of explanation.

We first provide some intuition about the framework and describe it formally in Section 3. Then we present in Section 4 a taxonomy of black-box explanation systems derived from our framework and describe the instantiation of the framework to an example of explanation method. We illustrate the interest of the framework for the design of explanation systems in Section 5. Finally, we provide an overview of related work in Section 6 and conclude with some perspectives and future work in Section 7.

3 Description of the framework

We first provide an overview and some intuition about our framework in Section 3.1, before presenting the Sampling and Generation components more formally in Section 3.2 and Section 3.3 respectively.

3.1 Overview

To introduce our framework we consider the concrete example of a spam classifier. The system takes as input the text of an email and outputs the probability of this email being a spam. Ideally, an explanation system (herefater, "explainer") should be able to answer a wide range of questions because different explainees have different interests, motivations and levels of expertise. For example, a user of the spam classifier may want to ask questions to better understand the system or its behavior in specific circumstances. Possible questions on his part include "Did the signature part of email $\mathbf{x}^{(e)}$ have an impact on the fact that it has been classified as a spam?" or "Why is email $\mathbf{x}^{(e)}$ classified as a spam and not email $\mathbf{y}^{(e)}$?" The explanations can be useful to enhance his trust in the classifier or to allow him to understand how to modify its parameters if it does not behave as expected. On the other hand, the designer of the system may have more precise requests such as are "What are the main features used by the classifier to decide that an email is likely to be a spam and what are their respective weights?" Since we assume that the code of the classifier is not available, the explainer can only build emails, submit them to the classifier and analyze the results. For example, to answer the first question of the user, the explainer can create different versions of $\mathbf{x}^{(e)}$ with and without the signature part, or with different pieces of text in the signature part. The explainer has then to evaluate the answer based on the results of the classifier and to present it to the explainee.

This simple example highlights the three main tasks of an explainer which are pictured in Figure 1 : (i) the *Interaction* task, which includes the input and analysis of the questions of the explainee and the presentation of the explanations in an intelligible way; (ii) the selection of inputs to submit to the system to be explained, which is called the *Sampling* task; and (iii) the analysis of the links between the selected inputs and the corresponding outputs of the system to generate the content of the explanations, which is called the *Generation* task. In many cases, the *Sampling* task and the *Generation* task are applied sequentially but it is sometimes useful to apply them iteratively, to be able to adjust the set of samples to the needs of the *Generation* task. As stated in the introduction, the *Interaction* task has not received as much attention as the two other tasks in the literature so far. Therefore, for the sake of conciseness we focus on the *Sampling* and the *Generation* tasks in this paper. We propose formal characterizations of these tasks which are generic enough to encompass existing proposals and to compare them on a rigorous basis, as discussed in Section 4 and sketched in Table 1.

3.2 Sampling

The role of the *Sampling* task is to select appropriate inputs (or "samples") to address a question of the explainee about a model M. The choice of the samples may depend on a number of factors. The first aspect to take into consideration is whether the question concerns the whole model or

Name	Description	Example
M	Black-box model	The spam classifier
I	Input space of M	Space of emails
O	Output space of M	$[0, 1]$
E	Scope of the explanation	Email $\mathbf{x}^{(e)}$
S	Samples (product of the sampling step)	Emails with changed signature
D	Dataset describing the overall population	Training set of M

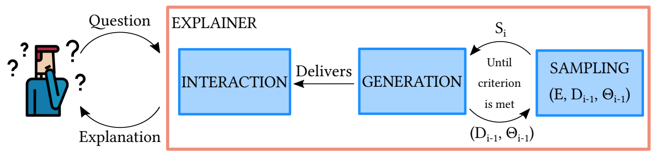

Figure 1: The three main tasks of the explaine

specific inputs. We call E the scope of the explanation. If the question concerns a single input $\mathbf{x}$, then $E = \{\mathbf{x}\}$; if the question is about the whole model M, then $E = D$ with D a representation of the population (all possible inputs to M) available to the explainer[1]. In general, E could be any subset of possible input values. We call I this set of input values, which can be seen as the support set (or type set) of D. In the spam filter example, I is the set of all possible emails and D represents the actual distribution of emails available to the explainer. In some cases, the explainer does not have any information about this distribution, which is denoted by $D = \emptyset$. The result of the *Sampling* task is a set of samples $S = \{\mathbf{x'}^{(1)}, ..., \mathbf{x'}^{(n)}\} \in I^n$. Different values of D may give rise to different sampling strategies. For example, to address the first question of the user of the spam filter about the impact of the signature on the classification of $\mathbf{x}^{(e)}$, a possible option is to select a single sample obtained by removing the signature part of $\mathbf{x}^{(e)}$. This strategy does not require any information about the actual distribution of the population and can therefore be applied with $D = \emptyset$. However the answer might not be realistic or precise enough because it would not provide any information about the way the content of the signature is actually taken into account by the filter. A more elaborate strategy would be to replace the original signature of $\mathbf{x}^{(e)}$ by real signatures obtained from many other emails. This strategy requires information about the actual distribution of the population ($D \neq \emptyset$) in order to ensure that the sample set is a reflection of the reality.

We can now define the sampling procedure as follows:

$$S = \{h_\theta(\mathbf{x}^{(e)}, \mathbf{x}^{(p)}) \mid (\theta, \mathbf{x}^{(e)}, \mathbf{x}^{(p)}) \in \Theta \times E \times D, F(\theta, \mathbf{x}^{(e)}, \mathbf{x}^{(p)}) = 1\} \tag{1}$$

with

$$h_\theta : E \times D \to I \tag{2}$$

Θ is the set of parameters for the sampling and F is a filter function. In a nutshell, the θ parameter makes it possible to generate several samples for a pair $(\mathbf{x}^{(e)}, \mathbf{x}^{(p)})$ while F makes it possible to generate samples only for a selection of pairs $(\mathbf{x}^{(e)}, \mathbf{x}^{(p)})$. In our spam filter example, E is limited to a single email to be explained ($E = \{\mathbf{x}^{(e)}\}$), there is no need for parametrization so we take $\Theta = \{0\}$ and we assume that D contains 1000 emails. Function $h_0(\mathbf{x}^{(e)}, \mathbf{x}^{(p)})$ returns

[1]It should be noted that D is actually a multiset since it can involve multiple occurrences of the same value to reflect the actual distribution of the values in the real population.

an email sample obtained from $\mathbf{x}^{(e)}$ by replacing its signature part by the signature part of $\mathbf{x}^{(p)}$. If F is the function that returns always 1 ($F(\theta, \mathbf{x}^{(e)}, \mathbf{x}^{(p)}) = 1$), then the sampling procedure generates 1000 perturbed version of $\mathbf{x}^{(e)}$ with signatures extracted from the emails in D. Another option could be to use a filter function F relying on a notion of distance and selecting only emails close to $\mathbf{x}^{(e)}$ or a function F selecting only emails with the same subject part as $\mathbf{x}^{(e)}$. The θ parameter can be used to customize the sampling function. For instance, if both the header and the signature of the email are taken into consideration, θ could specify which part of the email is replaced (header, signature or both).

3.3 Generation

When the set S of samples is available (or an element of S in the case of an iterative process), the next step consists in providing the elements of S as inputs to the model M and to collect the outputs : $\{\ (\mathbf{x'}, M(\mathbf{x'})) \mid \mathbf{x'} \in S\}$. This set is the raw materiel to build the explanations. Even if explanations can take many different forms, the *Generation* task can conceptually be split into two parts: the computation of a *proxy* of the model M and the construction of an explanation based on this proxy. We call the former *model generation* (G_M) and the latter *explanation generation* (G_E) in the sequel. In some cases, the proxy model is considered as the explanation itself, in which case no explanation generation is necessary; in other cases, the proxy model can be seen as an intermediate step to derive the explanation delivered to the explainee. We emphasize that we present a conceptual view of the *Generation* task here: the actual implementation of an explanation system does not necessarily involve the construction of the proxy model. Coming back to the spam classifier example, an option for the *Generation* task is to train a simple rule-based model on the samples resulting from *Sampling* task to predict the output of the classifier. An example of rule generated by this step could be: "If the signature of the email is less than 60 characters long, then the classifier will consider that it is a spam; otherwise it will be considered as an acceptable email". Because such rules are easily interpretable, they can directly be used as explanations. In other situations, either because the type of model used is too complex or the model is too big to be understandable (for example if it involves a large number of rules), simpler explanations have to be generated from the proxy model. This explanation generation phase can produce, for example, the most important feature(s) of the input. For the spam classifier, the answer in this case could be: "The length of the signature part and the number of typos are the two most important features used by the system to decide if an email is a spam".

Technically speaking, the proxy model is denoted by F_w, which is a function of the same type as the model M parameterized by w:

$$F_w : I \to O, \tag{3}$$

The core of the *Generation* task is to find the best F_w to answer the question of the explainee, which amounts to find the optimal values of w. Optimality can be defined formally using sets of criteria $o_i(w, S) \in \mathbb{R}$ and constraints $c_i(w, S) \in \mathbb{B}$ where $\mathbb{R}$ and $\mathbb{B}$ are the sets of real numbers and booleans respectively. The global objective takes the following form:

$$\begin{aligned} w^* = \operatorname*{argmin}_w \quad & \sum_i \lambda_i c_i(w, S) \\ \text{subject to} \quad & o_i(w, S) \end{aligned} \tag{4}$$

where $\lambda_i \in \mathbb{R}$ are used to weight the criteria. In many methods, the objective is to find the parameters w such that the proxy is as close as possible to M on the samples of S. However, using both criteria and constraints provides a great flexibility, which contributes to the generality of our framework. Finding a good explanation is often a question of compromise. A typical example is finding the right balance between precision and complexity – often used as a characterization of understandability. For example, a simple explanation of the spam classifier that would be accurate (i.e. predicting the actual result of the classifier) on only seventy percent of its inputs would not be acceptable; on the other hand, an accurate explanation that would take the form of several pages of rules would provide little insight to the user. As discussed in the following section, criteria and constraints can be used to define the priorities among objectives.

The second step of the of the *Generation* task, the explanation generation which delivers the explanation, is generally less technical. For instance, Shapley [26], PDP ICE [13] or VIN [11] compute sums of elements to produce a plot or specific numbers; LIME [18] extracts the coefficients of a linear function; LEMNA [8] and Local-gradients [1] derive from F_w slopes in the neighborhood E.

4 Taxonomy of black-box explanation systems

We first show in Section 4.1 how our generic framework can be used to analyze and classify existing explanation methods (Table 1). In view of space limitations, we cannot provide the details of the instantiation of the framework for each method of Table 1 but we present as an illustration the case of counterfactual explanations in Section 4.2. The interested reader can find the details of the definition of the methods Table 1 in our framework in in a longer version of this paper published as a research report [10].

4.1 Key features

The generic definitions introduced in the previous section allow us to highlight the range of choices in the design of a BEM and to classify existing methods based on these choices. Table 1 summarizes these design choices and the types of explanations produced by these methods. In this section, we provide some intuition about the table, considering successively the design choices related to the *Sampling* task, the *Generation* task and general choices (two columns to the left).

Design choices related to the *Sampling* task:

- The set E makes it possible to express the focus of the explanation: if the explainee is interested in a specific input data (e.g. an email $\mathbf{x}$) then E is a singleton set (e.g. $E = \{\mathbf{x}\}$). At the other end of the spectrum, the explainee may be looking for a global explanation of the system on the whole input domain ($E = D$). Ideally, the explainee should be able to choose any scope between these two extremes. However, as shown in Table 1, existing methods assume a fixed scope, which is almost always D or a singleton set. The only exception is MMD-Critic [12] that makes it possible to focus on a class of input data.

- In general, a model may behave differently on different segments of the population. For instance, it has been shown that face recognition systems are more accurate among white male than among non-white female [2]. Therefore, the fact that the distribution of the population is taken into account or not is an important feature of a BEM. This information is defined by set D in our framework. D is the information available to the BEM about the population *Pop*. Table 1 shows that all methods except LIME and LEMNA take the population into account. The population does not have any impact on the explanations produced by LIME and LEMNA ($D = \emptyset$) because the samples are obtained by random masking of the input data. The relevance of the use of the population depending on the question of the explainee is further discussed in Section 5.

- The "Type" column of Table 1 aggregates several pieces of information about the strategy used in the *Sampling* task. First, we make a distinction between *selection* sampling, which is characterized by the fact that samples necessarily belong to D ($h_\theta : E \times D \to D$) and *perturbation* sampling which can produce samples outside D ($h_\theta : E \times D \to I$). We also distinguish *deterministic* sampling, which always returns the same set of samples, and *random* sampling.

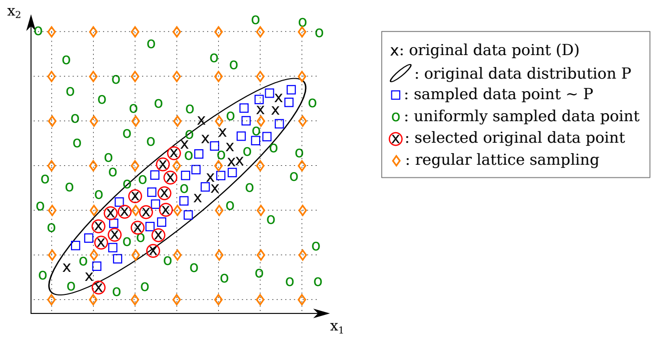
x₂
x₁
x: original data point (D)
: original data distribution P
: sampled data point ~ P
: uniformly sampled data point
: selected original data point
: regular lattice sampling

Figure 2: Illustration of sampling categories in 2 dimensions

Design choices related to the *Generation* task:

- Many BEM leverage existing machine learning methods to generate an explicit proxy F_w approximating M. The "Model" column of of Table 1 shows the type of "interpretable" model used by these BEMs (Trepan [3], BETA [14], Anchors [19], LIME [18], LEMNA [8], Local-Gradient [1]). Some of these methods return the interpretable model itself as an explanation while others apply a further explanation generation task, which is denoted by G_E in the "Steps" column.

- Column "Objectives" shows the criteria and constraints (respectively o_i and c_i in Definition 4 of Section 3.3) used by the BEM to compute F_w. Defining an objective as a constraint rather than a criterion is a way to assign it a higher priority. For instance, BETA [14] uses three constraints on the complexity of the rule-based proxy model and five criteria related to the fidelity of the proxy to M. Conversely, Anchors [19] sets a constraint on the fidelity of the rule-based model and criteria on the number of rules. As suggested by the titles of the papers, BETA focuses on interpretability of the explanation while Anchors puts more emphasis on fidelity.

General design choices:

- The "I/O" column in Table 1 provides information about the fact that the *Sampling* task is called iteratively (I) or is a one-shot task (O). Intuitively, the iterative mode may lead to sample sets that are more precise beacuse they are tailored to the needs of the *Generation* task. Selecting 1000 emails in the example of 3 is an example of one-shot mode. Another option could be to select only 100 samples in a first iteration step. A second iteration would focus on the region that was underrepresented in the first sample set, for example by querying only samples with long signatures. This strategy could be useful to reduce the number of model calls in the *Generation* task.

- To conclude, Column "Steps" characterizes each BEM based on the three steps identified in Section 3.2 and Section 3.3: respectively Sampling (S), model generation G_M and explanation generation (G_E). It is interesting to notice that six of the BEM in the table involve the tree steps, only one of them (Local Gradient) does not involve sampling, two of them do not have any model generation step and three of them do not have any explanation generation phase (they return the proxy model as an explanation). For example, Shapley [26] and PDP ICE [13] rely on specific rules to combine directly elements of S to build explanations, hence they bypass the model generation step.

Name	Sampling			Generation		Output Type	Steps[c]	I/O[d]
	E	D	Type[a]	Model	Objectives			
Trepan [3]	Pop.	Pop.	P&R	Decision tree	Fidelity, Complexity	Decision tree	S G_M	I
BETA [14]	Pop.	Pop.	S&D	Rule-based model	Fidelity, Unambiguity Interpretabilty	Rule-based global model	S G_M	I
GoldenEye [9]	Pop.	Pop.	P&R	Permutation	Fidelity, Complexity	Selection of important var.	S G_M G_E	I
VIN [11]	Pop.	Pop.	P&D	Permutation	ANOVA projection	Var. importance interactions	S G_M G_E	I
PDP ICE [13]	Pop.	Pop.	P&D	NA	NA	Plot	S G_E	O
MMD-critic [12]	Class	Pop.	S&D	NA	Distance to class, # prototypes	Example-based	S G_M G_E	O
Anchors [19]	$\{\mathbf{x}^{(e)}\}$	Pop.	P&R	Rule-based model	Fidelity, Complexity, Generality	Rule-based local model	S G_M	I
LIME [18]	$\{\mathbf{x}^{(e)}\}$	$\emptyset$	P&R	Linear model	Fidelity, Complexity	Var. importance	S G_M G_E	O
Shapley [26]	$\{\mathbf{x}^{(e)}\}$	Pop.	P&D	NA	NA	Var. importance	S G_E	O
LEMNA [8]	$\{\mathbf{x}^{(e)}\}$	$\emptyset$	P&D	Mixture of linear models	Fidelity, Complexity	Var. importance	S G_M G_E	O
Local Gradient [1]	Pop.	Pop.	S&D	Parzen window	Fidelity	Directions of highest slope	G_M G_E	O
Counter-factuals [24]	$\{\mathbf{x}^{(e)}\}$	Pop.	NA	Small deviation	Target output, Distance input	Example-based	S G_M G_E	I

Table 1: Comparative table of the different black-box explanation methods. The columns correspond to the parameters of our framework with the following notation (a) S: Selection, P: Perturbation, D: Deterministic, R: Random ; (b) G: Global, L: Local; (c) S: Sampling, G_M: model generation phase, G_E: explanation generation phase; (d) I: Indirect, O: One-shot.

4.2 Example: counterfactuals

Providing a counterfactual input to the user is an efficient way of explaining the output of an individual $M(\mathbf{x}^{(e)})$ in many situations. The explanation consists of an alternative input **x'** used as a point of comparison with $\mathbf{x}^{(e)}$, which is in line with the contrastive nature of explanations. More precisely, the version proposed here (inspired from [24]) looks for the closest alternative input that gives a different output: **x'** such that $M(\mathbf{x'}) = \mathbf{y'}$, with **y'** an alternative output s.t. $M(\mathbf{x}^{(e)}) \neq M(\mathbf{y'})$, while keeping $distance(\mathbf{x}^{(e)}, \mathbf{x}')$ small. In the spam classifier example, a good counterfactual is a slightly modified version of the email (e.g. removing one word) that would change the output of the model. It establish a causal relationship between the presence of the sole word and the classification as spam.

The computation of counterfactuals does not involve an approximation of the original model. Contrarily, it involves a generation function whose goal is to perturb M. More precisely, we can describe generation with the following function:

$$F_w(\mathbf{x}) = T_w(M(\mathbf{x})) = M(x + w) \tag{5}$$

where T_w denotes a translation operator s.t. $\mathbf{x} + w \in I$ and w the perturbation from the original input. Here, w does not represent the parameters of a model but a perturbation in the input space I. Interestingly, our framework is general enough to correctly describe both. The optimization problem is to find w^* such that:

$$\begin{aligned} w^* = \underset{w}{\text{argmin}} \quad & distance(\mathbf{x}^{(e)}, \mathbf{x}^{(e)} + w) \\ \text{subject to} \quad & F_w(\mathbf{x}^{(e)}) = \mathbf{y'}, \end{aligned} \tag{6}$$

with **y'** an alternative output specified by the user. The generation of counterfactual is iterative. Samples are queried one-by-one until a stopping criteria is met. The resulting counterfactual $\mathbf{x}^{(e)} + w$ is returned as an explanation. In our previous example, the email with one word removed is delivered to the user.

Interestingly, many sampling strategies are practicable and they lead to different results. For example, if the user wants the counterfactual to be a real sample, a selection sampling should be considered. Otherwise, a random sampling strategy is employed. Using the population distribution leads to a close and likely samples while not using it leads to the closest sample even if the latter is unrealistic. For instance, if any mail containing "Fg_f" is classified as spam, then the generation step could explained any non-spam by adding this meaningless sequence of characters. On the other hand, sampling from the population distribution, could force the counterfactuals to be more realistic. User's motive should be considered in this choice. A lay user may be interested in realistic counterfactual while learning about this odd feature can be valuable to a system designer in the purpose of debugging the classifier.

5 Exploration of the design space

Beyond its interest to provide a systematic overview of existing explanation methods, as described in the previous section, our framework can provide valuable help for designers of new explanation systems. As suggested in the introduction, an explanation system should actually be interactive and allow explainees to ask different types of questions to enhance their understanding of the model. Depending on the type of questions, different choices can be made for the sampling and generation tasks.

As an illustration, we show in this section how the type of question should influence the sampling task. We use a simplified version of the previous spam classifier example in which M

has only one feature: the ratio of capital letters r_{cap}. The output remains the probability of being a spam. The shape of M is shown Fig. 3 (a). $P_{r_{cap}}$ denotes the probability distribution of r_{cap}, Fig. 3 (b). We compare a sampling from $P_{r_{cap}}$ with a uniform sampling between 0 and 1.

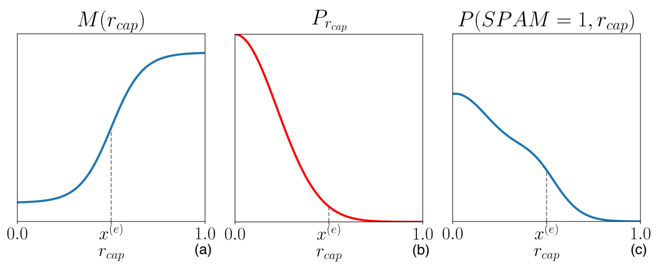

Figure 3: (a) Shape of the model $M(r_{cap})$ (b) Probability distribution of r_{cap} (c) Joint probability of $SPAM\!=\!1$ and r_{cap}

They respectively generate two S_p and S_u:

$$\begin{aligned} S_p =& \{r_{cap} \sim P_{r_{cap}}\} \\ =& \{0.03, 0.21, 0.17, 0.1, 0.4, 0.01, 0.0\} \end{aligned} \tag{7}$$

$$\begin{aligned} S_u =& \{r_{cap} \sim Unif(0,1)\} \\ =& \{0.45, 0.87, 0.15, 0.4, 0.98, 0.49, 0.54\} \end{aligned} \tag{8}$$

As expected, samples from the population distribution are more concentrated near small values. We focus on a local explanation: $E = \{\mathbf{x}^{(e)} = (0.5)\}$.

We rely on the work of [21] to address the first type of question: "Did this value of r_{cap} influenced the model toward spam or toward non-spam?". A way to answer is to estimate what would be the output of the model in the absence of the knowledge of r_{cap}. To do so, we can compare the current value with an average situation, which suggests the use of the population distribution. Because $(M(\mathbf{x}^{(e)}) - \langle M(\mathbf{x'})\rangle_{\mathbf{x'}\in S_p}) > 0$, the output of the model would have been lower in the absence of the knowledge of r_{cap}. Therefore, this value contributed toward the classification as spam. It should be noticed that the result would have been different if the uniform distribution were used instead, indeed $(M(\mathbf{x}^{(e)}) - \langle M(\mathbf{x'})\rangle_{\mathbf{x'}\in S_u}) \approx 0$.

The second type of question is: "What would be the impact of an increase of r_{cap} on the prediction of the model?". In this case, the explainer needs to estimate the shape of M. To be efficient and to cover the full range of value, dispatching samples uniformly seems to be a good strategy. Consequently, for this type of question, the uniform distribution is preferable. Contrarily to the first question, the population distribution would still give the good result but the generation would be less efficient and the result less precise.

To conclude this section, we give a probabilistic point of view of the problem. The goal of the model M is to estimate the conditional probability of being a spam knowing $\mathbf{x}^{(e)}$. Thus: $M(r_{cap}) = \mathbb{P}(SPAM\!=\!1 \mid r_{cap})$. If we imagine that samples are random variable (s.t $SPAM = 1$ with probability $M(r_{cap})$) then the underlying distribution of $M(S_u)$ is given by $\mathbb{P}(SPAM\!=\!1 \mid r_{cap})$, while the distribution of $M(S_p)$ would be the joint probability of observing a spam and this value of r_{cap}:

$$\mathbb{P}(SPAM\!=\!1 \mid r_{cap}) \times P_{r_{cap}} = \mathbb{P}(SPAM\!=\!1, r_{cap}) \tag{9}$$

(Fig. 3 (c)). We see that the shapes of the distributions are different.

6 Related work - A detailler

The present paper is not the first attempt to provide a unification in the field of XAI. Close to our subject, [16] introduces a formal theoretical framework to unify four BEMs. The framework is restricted to methods that compute the contribution of each feature for a given prediction. Moreover, it does not identify steps that are shared by any BEMs. With a broader approach, [7] introduce a glossary and a taxonomy for interpretable and explainable AI. It then exhaustively list all publications related to this field in the former taxonomy. Our work differs in two points: 1. we limit ourselves to the black-box assumption (no access to code or model), 2. we define a framework to describe BEM from which is deduced our taxonomy. At a higher level, [23] gives a taxonomy of machine learning components that may require explanation, interpretations, or more interaction. Our work situates in the ”Model” and ”Prediction” categories of their taxonomy. [20] provides a taxonomy of interpretability in Human-Agent Systems. Their interest is more general as it also includes the motive and the expected form of the interaction with the explainer. The answer to the ”What” question (section 3) is closely related to our subject. However, their work on the BEM is less detailed as they mention one method only. [22], [15] and [25] treat more generally of the needs in explainability and transparency regarding social and technical aspects. Finally, [4] provides a formal definition of explanations along with ways to evaluate them.

This attempt to unify BEMs in a general framework to go toward an explanation tool that adapts to the user’s question is, to our knowledge, new.

7 Conclusion - A ecrire

In this section, we now temper these advantages with some problems that we identified with our framework. The goal is to set the limits of this approach as well as suggesting improvements for future work. Even though the description of methods with the framework was possible in all tested cases, it can be sometime laborious. In particular Trepan [3] makes an estimation of the distribution of features instead of sampling directly from D. The benefits of this approach are arguable because estimating distribution is a complex task; it is harldy feasible with images for instance. The sampling of GoldenEye [9] involves mixing more than two inputs, which is not strictly described by our framework. If such sampling improves the detection of features interactions beyond the previous results of VIN [11], an extension of our framework can be considered.

Because we used current methods to make our assumptions, it is not impossible that a new method could not fit in our framework. For the sampling, our approach was to voluntary narrow as much a possible the type of methods that could be described. Our goal was not to provide a general framework for any sampling but to highlight the strategies that are the most common for generating explanations. For instance, because our framework is based on a one-to-one correspondence between elements (one element of $E \times D$ relates to one element of S), a uniform sampling without repetition could not be described.

Finally, the framework for generation is admittedly flexible. Since a wide variety of generation methods were encountered, it seemed preferable to allow the framework to evolve.

References

[1] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, and K. Hansen. How to explain individual classification decisions. page 29.

[2] J. A. Buolamwini. *Gender shades: intersectional phenotypic and demographic evaluation of face datasets and gender classifiers.* PhD thesis, Massachusetts Institute of Technology, 2017.

[3] M. Craven and J. W. Shavlik. Extracting tree-structured representations of trained networks. page 7.

[4] A. Dhurandhar, V. Iyengar, R. Luss, and K. Shanmugam. A formal framework to characterize interpretability of procedures. *arXiv:1707.03886 [cs]*, Jul 2017. arXiv: 1707.03886.

[5] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189âĂŞ1232, 2001.

[6] A. Goldstein, A. Kapelner, J. Bleich, and E. Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *arXiv:1309.6392 [stat]*, Sep 2013. arXiv: 1309.6392.

[7] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM Computing Surveys (CSUR)*, 51(5):93, 2018.

[8] W. Guo, D. Mu, J. Xu, P. Su, G. Wang, and X. Xing. LEMNA: Explaining Deep Learning based Security Applications. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security - CCS '18*, pages 364–379, Toronto, Canada, 2018. ACM Press.

[9] A. Henelius, K. PuolamÃďki, H. BostrÃűm, L. Asker, and P. Papapetrou. A peek into the black box: exploring classifiers by randomization. *Data Mining and Knowledge Discovery*, 28(5âĂŞ6):1503âĂŞ1529, Sep 2014.

[10] C. Henin and D. Le MÃľtayer. Towards a generic framework for black-box explanations of algorithmic decision systems (Extended Version). Inria Research Report 9276, https://hal.inria.fr/hal-02131174.

[11] G. Hooker. Discovering additive structure in black box functions. In *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD âĂŽ04*, page 575. ACM Press, 2004.

[12] B. Kim, R. Khanna, and O. O. Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems*, pages 2280–2288, 2016.

[13] J. Krause, A. Perer, and K. Ng. Interacting with Predictions: Visual Inspection of Black-box Machine Learning Models. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*, pages 5686–5697, Santa Clara, California, USA, 2016. ACM Press.

[14] H. Lakkaraju, E. Kamar, R. Caruana, and J. Leskovec. Interpretable & Explorable Approximations of Black Box Models. *arXiv preprint arXiv:1707.01154*, 2017.

[15] Z. C. Lipton. The mythos of model interpretability. *arXiv:1606.03490 [cs, stat]*, Jun 2016. arXiv: 1606.03490.

[16] S. Lundberg and S.-I. Lee. A Unified Approach to Interpreting Model Predictions. *arXiv:1705.07874 [cs, stat]*, May 2017. arXiv: 1705.07874.

[17] T. Miller. Explanation in artificial intelligence: insights from the social sciences. *arXiv preprint arXiv:1706.07269*, 2017.

[18] M. T. Ribeiro, S. Singh, and C. Guestrin. ”Why Should I Trust You?”: Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pages 1135–1144, San Francisco, California, USA, 2016. ACM Press.

[19] M. T. Ribeiro, S. Singh, and C. Guestrin. Anchors: High-precision model-agnostic explanations. In *AAAI Conference on Artificial Intelligence*, 2018.

[20] A. Richardson and A. Rosenfeld. A survey of interpretability and explainability in human-agent systems. page 7.

[21] M. Robnik-Åăikonja and I. Kononenko. Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):589–600, 2008.

[22] R. Tomsett, D. Braines, D. Harborne, A. D. Preece, and S. Chakraborty. Interpretable to whom? a role-based model for analyzing interpretable machine learning systems. *CoRR*, abs/1806.07552, 2018.

[23] E. Ventocilla, T. Helldin, M. Riveiro, J. Bae, and N. Lavesson. Towards a taxonomy for interpretable and interactive machine learning. 2018.

[24] S. Wachter, B. Mittelstadt, and C. Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *SSRN Electronic Journal*, 2017.

[25] A. Weller. Challenges for transparency. *arXiv:1708.01870 [cs]*, Jul 2017. arXiv: 1708.01870.

[26] E. Åătrumbelj and I. Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41(3):647âĂŞ665, Dec 2014.

The icon of Fig. 1 was made by turkkub from www.flaticon.com

Appendices

Unless specified differently, we assume that elements of the input space I are vectors in S dimensions composed of numerical and categorical values:

$$\mathbf{x} = (x_1, ..., x_K) \in I \tag{10}$$

A LIME

Local Interpretable Model-agnostic Explanations (LIME) was created in [18]. Basically, it fits a linear model "in the neighborhood" of the point $\mathbf{x}^{(e)}$ to be explained and uses the coefficients of the regression to estimate which features were the most influential for the model.

Given $\mathbf{x}^{(e)}$, it creates a dataset S composed of n "perturbated" replications of $\mathbf{x}^{(e)}$ by randomly setting s (random variable) features to 0. For instance with $K = 4$ and $n = 3$:

$$\begin{aligned} \mathbf{x}^{(e)} = (x_1, x_2, x_3, x_4) &\xrightarrow{s=1} \mathbf{x'}^{(1)} = (0, x_2, x_3, x_4) \\ &\xrightarrow{s=2} \mathbf{x'}^{(2)} = (x_1, 0, x_3, 0) \\ &\xrightarrow{s=2} \mathbf{x'}^{(3)} = (x_1, x_2, 0, 0) \end{aligned} \tag{11}$$

we create $D' = \{\mathbf{x'}^{(1)}, \mathbf{x'}^{(2)}, \mathbf{x'}^{(3)}\}$. Then, the output of each "perturbated" replications is computed and a weighted lasso regression is fitted on the resulting training set $\{(\mathbf{x'}, M(\mathbf{x'})) \mid \mathbf{x'} \in S\}$. The weights of the regression are decreasing as a function of the distance to $\mathbf{x}^{(e)}$: $exp(-d(\mathbf{x}^{(e)}, \mathbf{x'})^2/\sigma^2)$, with σ a parameter of the model , which ensure the locality of the explanation. Finally, the coefficients of the linear model are directly used to measure the importance of feature in the prediction of $\mathbf{x}^{(e)}$. Please refer to the original paper [18] for a more detailed description of the method.

Sampling:

$$E = \{\mathbf{x}^{(e)}\} \tag{12}$$

$$D = \emptyset \tag{13}$$

Interestingly, LIME is one of the only methods that do not use a dataset apart from the point to be explained $\mathbf{x}^{(e)}$. Firstly, we build the range of parameters:

$$\Theta = \left\{ \theta_j = \left\{ \text{SRSWOR of size } s_j \text{ in } \{1, ..., K\} \right\} \mid j \in \{1, ..., K\} \right\} \tag{14}$$

with $s_j \sim Unif\{0, K\}$, SRSWOR denotes "Simple Random Sampling Without Replacement" and $Unif\{0, K\}$ denotes the discrete uniform distribution of integers between 0 and K. θ_j is the set of features' index that are nullified. Now we can define the sampling functions.

$$h_\theta(\mathbf{x}_e, \mathbf{x}_p) = \left(\begin{cases} 0 & \text{if } j \in \theta \\ x_j & \text{otherwise} \end{cases} \right)_{j=1,\ldots,K} \tag{15}$$

h_θ acts like a masking function, it masks all features which index appears in θ. In (11), $\theta_1 = \{1\}$, $\theta_2 = \{2, 4\}$ and $\theta_3 = \{3, 4\}$. Then the equation (1) can be used to generate the database of perturbated examples as it is done in LIME. We can see that the sampling is a **perturbation** and **random**.

It should be noted that in the case of images the features are not set to 0 but to the value corresponding to the color grey.

Generation:

LIME uses F_w as an approximative model for M. More precisely, F_w is a linear function of the inputs:

$$F_w(\mathbf{x}) = \sum_{i=1}^{K} w_i x_i + w_0 \tag{16}$$

Two criteria are optimized during the generation (both as objective so we note o_1 and o_2).

$$o_1(w, S) = \sum_{\mathbf{x'} \in S} e^{-\frac{d(\mathbf{x}^{(e)}, \mathbf{x'})^2}{\sigma^2}} (F_w(\mathbf{x'}) - M(\mathbf{x'}))^2 \tag{17}$$

$$o_2(w) = \|w\| \tag{18}$$

c_1 ensures that the approximate linear model is faithful to M in the neighborhood of $\mathbf{x}^{(e)}$ while c_2 ensures that the resulting local model is not to complex. How should be weighted this two objectives is not clearly mentioned in the original article.

Delivering:

Once the model has been fitted, the coefficients of the linear regression are directly delivered to the user. It should be noted that in the case of image classification, it is more understandable to deliver to the user the pixels that have a positive value (pixels that contributed toward the prediction of the correct class).

B Anchors

Anchors [19] looks for the biggest square region of the input space that contains the initial input and which preserves the same output with high-precision. The explanation is a rule based model faithful locally around the initial input $\mathbf{x}^{(e)}$. It works on an iterative basis. At each step i, the analyzer proposes a candidate set of rules A_i, for example:

$$A_0 = \text{"}3 \geq x_1 \geq 2\text{" AND "}x_3 = 0.5\text{"}, \tag{19}$$

and the rules are tested by querying many samples that verifies it.

Sampling:

$$E = \{\mathbf{x}^{(e)}\} \tag{20}$$

$$D = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}\} \tag{21}$$

Anchors works with different types of data, which have different sampling strategies. However, the same idea remains. The parameters of the sampling θ contains the list of features that appear in the candidate set of rule as current iteration. For example, for the anchor defined by (19), the corresponding parameters are $\theta_0 = \{1, 3\}$. Samples mix features of $\mathbf{x}^{(e)}$ and $\mathbf{x}^{(e)}$. Features that appear in θ are taken from $\mathbf{x}^{(e)}$ and others from $\mathbf{x}^{(p)}$. For tabular and image data the expression of h is the same:

$$h_\theta(\mathbf{x}_e, \mathbf{x}_p) = \left(\begin{cases} x_j^{(e)} & \text{if } j \in \theta \\ x_j^{(p)} & \text{otherwise} \end{cases} \right)_{j=1\ldots K} \tag{22}$$

For text classification, the sampling is more elaborate. If $\mathbf{x} \in I$ is sequence of words, then x_j denotes the j^{th} word in the sentence. θ still defines the candidate set of rules and its elements are the words that it contains. Then:

$$h_\theta(\mathbf{x}_e, \mathbf{x}_p) = \left(\begin{cases} x_j^{(e)} & \text{if } x_j^{(e)} \in \theta \\ x_k^{(p)} & \text{with proba } p \text{ otherwise} \\ x_j^{(e)} & \text{with proba } (1-p) \text{ otherwise} \end{cases} \right)_{j=1\ldots|\mathbf{x}_e|} \tag{23}$$

with $x_k^{(p)}$ such that $x_k^{(p)}$ and $x_j^{(e)}$ have the same POS tag. And with p proportional to their similarity in an embedding space. In both cases, the parameter space only contains one element $\Theta = \{\theta\}$

The optimization method employed by Anchors to minimize the number of model call requires that the samples are generated one-by-one (for the case of precision estimation, see below). The sample is generated with the parameter θ, the input to be explained $\mathbf{x}^{(e)}$ and one sample from D selected randomly. The random selection of a sample is made thanks to the filter function F.

$$F(\theta, \mathbf{x}_e, \mathbf{x}_p) = \mathbb{1}_{\mathbf{x}_p = \mathbf{x}^{(idx)}} \tag{24}$$

with $idx = Unif\{1, N\}$ (drawn at each new sampling). The stochastic nature of the sampling of Anchors comes from this drawing.

Generation:

Anchors is a typical generation case. F_w is an interpretable model used to approximate M. More precisely, F_w is a rule-based model (RBM) and the rules are represented by w.

$$F_w = RBM_w \tag{25}$$

The goal of the optimization is to find the rules that cover the most examples while satisfying probabilistic fidelity constraints. To evaluate the coverage, an extra sampling step is required with $\theta = \emptyset$ ($\Theta = \{\emptyset\}$), we call the result of the sampling $S_\emptyset$.

$$o(w, D) = -\frac{\sum_{\mathbf{x'} \in S_\emptyset} \mathbb{1}_{\mathbf{x'} \text{ verifies } w}}{|S_\emptyset|} \tag{26}$$

And the constraint on the fidelity can be expressed as:

$$c(w, D) = \mathbb{P}\Big(\frac{\sum_{\mathbf{x'} \in S} \mathbb{1}_{F_w(\mathbf{x'}) = M(\mathbf{x'})}}{|S|} \geq \tau \Big) \geq (1 - \delta) \tag{27}$$

In the theoretical formulation, the size of S is arbitrary. The optimization problem can be formulated as in (4). The optimization process is made iteratively. Each step of Anchors consists of 2 iterations in our framework: one iteration for each coverage estimation and one iteration for each precision estimation.

Delivering:

Anchors does not require a delivering step as the explanation is the model F_w itself.

C Shapley values

In [26], the authors are using Shapley values, a result from cooperative game theory, to build a measure the contribution of a feature for a specific classification (local explanation). Shapley's result is interesting because it has four mathematical properties which ensure consistency with what is expected of a feature contribution. We note $\mathbf{x}^{(e)}$ the point to be explained, as in [26] we can define for any $Q \subseteq \{1, ..., K\}$

$$M_Q(\mathbf{x}^{(e)}) = \mathbb{E}\Big[M(X_1, ..., X_K) \mid X_i = (\mathbf{x}^{(e)})_i, \forall i \in Q\Big] \tag{28}$$

$$\Delta_Q(\mathbf{x}^{(e)}) = M_Q(\mathbf{x}^{(e)}) - M_{\{\}}(\mathbf{x}^{(e)}) \tag{29}$$

Then, the importance of the variable i for the data point $\mathbf{x}^{(e)}$ is given by the Shapley value:

$$\phi_i(\mathbf{x}^{(e)}) = \sum_{Q \subseteq [1,...,K] \setminus \{i\}} \frac{|Q|!(K - |Q| - 1)!}{K!} \Big(\Delta_{Q \cup \{i\}}(\mathbf{x}^{(e)}) - \Delta_Q(\mathbf{x}^{(e)})\Big) \tag{30}$$

Please refer to the original paper for more details on the meaning of this formula.

Sampling:

$$E = \{\mathbf{x}^{(e)}\} \tag{31}$$

$$D = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}\} \tag{32}$$

The parameters of the sampling θ contains the list of features that appear in the tested combination Q: $\theta = Q$. Samples mix features of $\mathbf{x}^{(e)}$ and $\mathbf{x}^{(e)}$. Features that appear in θ are taken from $\mathbf{x}^{(e)}$ and others from $\mathbf{x}^{(p)}$.

$$h_\theta(\mathbf{x}_e, \mathbf{x}_p) = \left(\begin{cases} x_j^{(e)} & \text{if } j \in \theta \\ x_j^{(p)} & \text{otherwise} \end{cases} \right)_{j=1...K} \tag{33}$$

Then, we define the range of parameters:

$$\Theta = \Big\{Q \mid Q \subseteq \{1, ..., K\}\Big\} \tag{34}$$

We can see that the sampling if a **deterministic perturbation** of the inputs.

Generation:
There is no generation step for this method.

Delivering:
We now assume that (28) is best approximated by the empirical mean:

$$M_Q(\mathbf{x}^{(e)}) \approx \frac{1}{|S|} \sum_{\mathbf{x'} \in S} M(\mathbf{x'}) \tag{35}$$

with S defined by (1). We can now see that the Shapley value can be approximated by averaging the dataset defined by (1), with the weights of (30). The approach is **direct**.

D PDP and ICE

Partial Dependence Plot (PDP) was introduced in [5] and Individual Conditional Expectation (ICE) in [6] and both were revised in the lens of explanation in PDP ICE [13] (among others). The version that we present here in the latter.

It is a feature-wise global explanation method to estimate the global impact of a feature i over its range of values. The output is a plot whose x-axis is all possible values of i and y-axis the average value of the model M over S with the value of the i^{th} fixed (PDP) or the superposition of every sample of S with modified value for i (ICE).

In both cases, the sampling is a **deterministic perturbation** over the full range of values. This method is a global explanation method in which $E = D$.

Sampling:

$$E = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}\} \tag{36}$$

$$D = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}\} \tag{37}$$

As mentioned in the paper, it can be more convenient to use only value of E, which is the case here. Hence, in the following, we artificially set $D = 0$. It should be noted that it is not incompatible with our theoretical framework as it is always possible to use F to filter unneeded elements.

Since the explanation is feature-specific, we can assume that each BEM explain one feature i. If the i is a continuous value, we can divide its range in r regular steps.

$$l_i = \min_{\mathbf{x} \in I} x_i, \quad u_i = \max_{\mathbf{x} \in I} x_i \tag{38}$$

$$h_\theta(\mathbf{x}_e, 0) = \left(\begin{cases} l_i + \theta \dfrac{u_i - l_i}{r - 1} & \text{if } j = i \\ x_{e,j} & \text{otherwise} \end{cases} \right)_{j=1...K} \tag{39}$$

$$\Theta = \{0, 1, ..., r - 1\} \tag{40}$$

If the feature i is categorical, we note $\{v_1, ..., v_r\}$ the possible values:

$$h_\theta(\mathbf{x}_e, 0) = \left(\begin{cases} \theta & \text{if } j = i \\ x_j & \text{otherwise} \end{cases} \right)_{j=1...K} \tag{41}$$

$$\Theta = \{v_1, ..., v_r\} \tag{42}$$

We can now generate the sample set S using (1).

Generation:

There is no generation step for this method.

Delivering: The explanation can be directly computed from the set $\{(\mathbf{x'}, M(\mathbf{x'}))\mathbf{x'} \in S\}$ by attributing to the first value of Θ the average of the first k element of the previous set (PDP) or by plotting directly each value.

E BETA

Black box Explanation Through transparent Approximation (BETA) [14] is a global explanation method that approximate the model by an interpretable rule-based model. Selected model tries to minimize 8 criteria through an optimization procedure. The optimization is **iterative** and a candidate rule-set is evaluated at each iteration.

Sampling:

$$E = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}\} \tag{43}$$

$$D = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}\} \tag{44}$$

As mentioned in the paper, for some global methods, it can be more convenient to use only value of E, which is the case here. Hence, in the following, we artificially set $D = 0$. It should be noted that it is not incompatible with our theoretical framework as it is always possible to use F to filter unneeded elements.

The sampling parameters θ define a candidate set of rules. For ease of notation, we use $\forall textbfx \in I$, $\theta(\mathbf{x})$ is boolean value (true if $\mathbf{x}$ verifies the rule defined by θ false otherwise). Then the sampling is defined by:

$$h_\theta(\mathbf{x}_e, \mathbf{x}_p) = \mathbf{x}_e \tag{45}$$

$$F(\theta, \mathbf{x}_e, \mathbf{x}_p) = \mathbb{1}_{\theta(\mathbf{x}_e)} \tag{46}$$

The sampling of BETA is **selective** and **deterministic**.

Generation:

The generation function is an approximative model which tries to maximize the fidelity with M and other criteria. The approximative model is a rule based model: $F_w = RBM_w$ and the criteria to optimized are explicitly mentioned in the article (cf section 2.2 of [14]). One can simply verify that every criteria appearing in the optimization problem of BETA (cf. equations (1) and (2) of [14]) can be expressed analytically w.r.t w and D (in our case E).

Delivering:

There is no need for a delivering step as the rule-based model itself is provided as an explanation.

F LEMNA

”Local Explanation Method using Nonlinear Approximation” (LEMNA) [8] is a local explanation method. It is based on the framework as LIME but uses a mixture regression model with a fused Lasso regularization.

Sampling:

The sampling part of LEMNA is similar to LIME as it is described in A. [8] does not give a lot of details on the sampling procedure used by their method. The sentence ”The idea is to randomly nullify a subset of features of x.” suggest that the sampling function used is defined as in (15). But the absence of weights decreasing with the distance to $\mathbf{x}^{(e)}$ and the sentence ”we first synthesize a set of data *samples locally (around x)*” suggest that the choice of sampling parameters enforce locality.

Generation:

The generation step is based on an approximation of M in the neighborhood of the data point to be explained $\mathbf{x}^{(e)}$. The generation function is a mixture of linear models (MLM) $F_w = MLM_w$. Similarly to LIME, the criteria to optimize are the fidelity of the approximation model and a regularization:

$$o_1(w, S) = \sum_{\mathbf{x'} \in S} (F_w(\mathbf{x'}) - M(\mathbf{x'}))^2 \tag{47}$$

$$o_2(w) = \|w\|_{fussed} \tag{48}$$

The fidelity criterion is the traditional one but the regularization is tailored for application with data having a sequential structure (please refer to the equation (7) of the original article for more details).

Delivering:

As for LIME A, the coefficients of the linear regression model are used as explanation. Unfortunately, few information is provided on which coefficients should be chosen, among all linear models.

G VIN

Variable Interaction Network (VIN) [11] is a global explanation method based on the theory of Analysis of Variance (ANOVA) which aims at detecting additive structure in a black-box. Two features are additive if they do not interact for the prediction. As Shapley values, VIN combines partial permutations of the samples to detect the influence of specific features or groups of features. But thanks to a clever formulation of the problem, VIN does not require a number of model estimations exponential in the number of features which make it practical for more applications.

The VIN algorithm uses a iterative approach. While evaluating the importance of some combinations of features, VIN gather formal guaranties of smallness for other combinations and can thus avoid the computation of these. As for VIN evaluate one-by-one the importance of combinations of variables. It uses a (rather complex) analytical formula but we show that all terms of this formula can be obtained thanks to our framework. The quantity of interest is a sum of terms of the form:

$$\mathbb{E}_{-v}[M(x)] = \frac{1}{N} \sum_{i=1}^{N} M(x_v, x_{i,-v}) \tag{49}$$

where $v \in \{1, ..., K\}$, $x_v = (x_i | i \in v)$ and $x_v = (x_i | i \notin v)$.

Sampling:

$$E = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}\} \tag{50}$$

$$D = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(N)}\} \tag{51}$$

The parameters of the sampling θ contains the list of features that appear in the tested combination v: $\theta = Q$. Samples mix features of $\mathbf{x}^{(e)}$ and $\mathbf{x}^{(e)}$. Features that appear in θ are taken from $\mathbf{x}^{(e)}$ and others from $\mathbf{x}^{(p)}$.

$$h_\theta(\mathbf{x}_e, \mathbf{x}_p) = \left(\begin{cases} x_j^{(e)} & \text{if } j \in \theta \\ x_j^{(p)} & \text{otherwise} \end{cases} \right)_{j=1...K} \tag{52}$$

Generation:

The generation function of the VIN algorithm is a perturbation of M by permutation of features of input as mentioned in (53). The rest of the generation only involves summation and iterative search over a lattice of subset. We invite the interested read to refer to [11] for more details. We do not give the details of the computation of the projection of M, instead we show that the basic component can be obtain with the following generation function:

$$F_w(\mathbf{x}) = \frac{1}{|S'|} \sum_{\mathbf{x} \in S'} M(\mathbf{x}) \tag{53}$$

with $S'(\mathbf{x}, w) = \{h_w(\mathbf{x}, \mathbf{x}_p) \mid \mathbf{x}_p \in D\}$ (derived from 1).

H MMD-critic

TODO

I Local-gradient

TODO

J Trepan

TODO

K GoldenEye

TODO

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399