Factual and Counterfactual Explanations for Black-Box Decision Making

Riccardo Guidotti, Anna Monreale, Fosca Giannotti, Dino Pedreschi, Salvatore Ruggieri, and Franco Turini

***Abstract*—The rise of sophisticated machine learning models has brought accurate but obscure decision systems, which hide their logic, thus undermining transparency, trust, and the adoption of AI in socially sensitive and safety-critical contexts. We introduce a local rule-based explanation method providing faithful explanations of the decision made by a black-box classifier on a specific instance. The proposed method first learns an interpretable, local classifier on a synthetic neighborhood of the instance under investigation, generated by a genetic algorithm. Then it derives from the interpretable classifier an explanation consisting of a decision rule, explaining the factual reasons of the decision, and a set of counterfactuals, suggesting the changes in the instance features that would lead to a different outcome. Experimental results show that the proposed method outperforms existing approaches in terms of the quality of the explanations and of the accuracy in mimicking the black-box.**

***Index Terms*—Explainable AI, Interpretable Machine Learning, Open the Black Box, Explanation Rules, Counterfactuals**

I. Introduction

Newspapers are full of commentaries about algorithms taking critical decisions that heavily impact on our life and society, from loan concession in bank systems to pedestrian detection in self-driving cars. The worry is not only due to the increasing automation of decision making, but mostly to the fact that the algorithms are opaque and their logic unexplained. The main cause of this lack of transparency is that decision-making algorithms are generated often from data through Machine learning (ML). ML allows building predictive models that map user features into a decision, obtained by generalizing from a dataset of examples. The process of inferring a classification model from examples cannot be easily controlled because the size of training data and the complexity of the learned model are too big for humans. The inability to obtain an explanation for a decision is a profound drawback of learning from data, limiting social acceptance and trust in its adoption in many sensitive contexts.

In this paper we address the problem of explaining the decision outcome taken by an algorithm providing "meaningful explanations of the logic involved" when automated decision making takes place, as prescribed by the "right to explanation" of the European General Data Protection Regulation. We perform our research under some specific assumptions. First, we assume that an explanation is interesting if it clarifies why a *specific decision* has been made, i.e., we aim for *local* explanations, not general, *global*, descriptions of how the overall system works [1], [2]. Second, we assume that the explanations should be as close as possible to the language of reasoning, which is formal logic. Thus, we assume the user can understand elementary logic rules, but it should also be considered that from logic rules it is easy to construct narratives that are understood by users with diverse expertise. Finally, we assume that the black-box decision system can be queried as many times as necessary, to probe its decision behavior to the scope of reconstructing its logic; this is certainly the case in a legal argumentation in court, or in an industrial setting. On the other hand, we make no assumptions on the algorithms used in the obscure classifier: we aim at an *agnostic* explanation method analyzing the input-output behavior of the black-box, disregarding its internals [3].

We propose LORE, a LOcal RUle-based Explanation method for tabular data. Given a black-box binary predictor b and a specific instance x labeled with outcome y by b, we build a simple, interpretable predictor by first generating a balanced set of neighbor instances of the given x through an ad-hoc genetic algorithm, and then extracting from such a set labelled with b a decision tree classifier. A *local explanation* is then extracted from the obtained decision tree. The local explanation is a pair composed by *(i)* a – factual – *logic rule*, corresponding to the path in the tree that explains why x has been labeled as y by b, and *(ii)* a set of *counterfactual rules*, explaining which changes in x would invert the class y assigned by b. For example, from the `compas` dataset we may have the following explanation: the rule $\{age{\leq}39, race{=}African\text{–}American, recidivist{=}True\}{\rightarrow}HighRisk$ and the counterfactuals $\{age{>}40\}$ and $\{race{=}White\text{–}American\}$. Here, the factual explanation is that the high risk of recidivism is predicted for a black younger than 40 with prior recidivism; the counterfactuals explain that a lower risk would be predicted if the person were either older than 40 or white. The usefulness of the explanation depends on the stakeholder: it may make sense to a judge that wants to understand and evaluate the suggestion by the decision support system and possibly discover that it biased against blacks.

The intuition behind our method, common to other *local* approaches, such as LIME [3], and ANCHOR [4] is that the decision boundary for the black-box can be arbitrarily complex over the whole data space, but in the neighborhood of a data point there is a high chance that the decision boundary is clear and simple, hence amenable to be captured by an interpretable model. These methods are named *local* because they focus on the behavior of the black-box in the neighborhood of the

Riccardo Guidotti and Fosca Giannotti are with ISTI CNR, Pisa, Italy. e-mail: firstname.lastname@isti.cnr.it.

Anna Monreale, Dino Pedreschi, Salvatore Ruggieri and Franco Turini are with University of Pisa, Italy . e-mail: firstname.lastname@unipi.it.

Manuscript received August ? , 2019; revised ?.

specific instance x, without providing a single description of the logic of the black-box for all possible instances. On the other hand, global methods like [2], aims at retrieving explanations for the whole logic of the model. The novelty of our method is twofold. First, the high expressiveness of the proposed explanation surpasses state-of-the-art methods providing not only succinct evidence why an instance has been assigned a specific label, but also counterfactuals suggesting what should be different in the vicinity of the instance to reverse the predicted outcome. In other words, our inferred explanations are both factual and counterfactual, in line with the cognitive psychology literature maintaining that counterfactuals help people to reason on explanations that identify cause-effect or reason-action relations between events [5], [6]. Similarly, [2] produces *"balanced"* explanations supporting and opposing to a fact. Second, the local decision boundary in the neighborhood of the instance to explain is explored through a focused genetic algorithm, which produces high-quality training data to learn the local decision tree.

We propose extensive experiments to assess the goodness of our explanation method with respect to existing linear, rule-based, and counter-factual-based explanation approaches.

II. Local Rule-based Explanation Method

Given the black-box b, and an instance x in the feature space $\mathcal{X}^{(m)}$, we aim to solve the *black-box outcome explanation problem* which consists in providing an explanation e for the decision $b(x) = y$. We assume that some knowledge is available about the feature space $\mathcal{X}^{(m)}$, i.e., the empirical distribution of the m features. Nothing is assumed about the process of constructing the black-box b.

As a solution to the black-box outcome explanation problem we propose LORE, a LOcal Rule-based Explanation method. LORE learns an interpretable predictor c that reproduces and accurately mimes the local behavior of b in the neighborhood Z of x. The neighborhood Z is generated by LORE as part of the explanation process through a *genetic algorithm* in order to accurately explore the local decision boundary of b. The synthetic instances in Z are then labeled using b, and this set is used to train an interpretable local predictor c. LORE adopts a *decision tree* as interpretable predictor c. Finally, a *factual and counter-factual explanation* e for the decision $b(x) = y$ is derived from the structure of c. The explanation consists of a decision rule r and a set of counterfactual rules Φ. Details are discussed in the rest of this section.

A. Factual and Counter-Factual Explanation

We define an explanation e as a pair of objects: $e = \langle r, \Phi \rangle$, where $r = p \rightarrow y$ is a *factual* decision rule describing the reason for the decision value $y = b(x)$, while Φ is a set of *counterfactual* rules, namely rules describing the minimal number of changes in the feature values of x that would change the decision of the predictor to $y' \neq y$. Given the instance $x = \{(age{=}22), (job{=}clerk), (income{=}800), (car{=}no)\}$, we consider the following explanation for a loan request:

$$e = \langle r = \{age{\leq}25, job{=}clerk, income{\leq}900\} \rightarrow deny,$$
$$\Phi = \{(\{income{>}900\} \rightarrow grant),$$
$$(\{job{=}employer\} \rightarrow grant)\}\rangle$$

In a factual decision rule r of the form $p \rightarrow y$, the decision y is the *consequence* of the rule, while the *premise* p is a boolean condition on feature values. We assume that p is a conjunction of split conditions of the form $a_i{\in}[v_i^{(l)}, v_i^{(u)}]$, where a_i is a feature and $v_i^{(l)}, v_i^{(u)}$ are lower and upper bound values in the domain of a_i extended with $\pm\infty$. An instance x *satisfies* r, or r *covers* x, if the boolean condition p evaluates to true for x, i.e., if the split conditions $sc(x)$ is true for every condition in p. The rule r in the example above is satisfied by $x = \{(age{=}22), (job{=}clerk), (income{=}800), (car{=}no)\}$. When the instance x for which we have to explain the decision satisfies p, the rule $p \rightarrow y$ represents *a motivation for taking* the decision, i.e., p explains the fact why b returned y.

Consider now a set δ of split conditions. We denote the update of p by δ as $p[\delta]{=}\delta \cup \{(a \in [v_i^{(l)}, v_i^{(u)}]) \in p |\ \nexists w_i^{(l)}, w_i^{(u)}. (a{\in}[w_i^{(l)}, w_i^{(u)}]){\in}\delta\}$. Intuitively, $p[\delta]$ is the logical condition p with ranges for attributes overwritten as stated in δ, e.g., $\{age{\leq}25, job{=}clerk\}[age{>}25]$ is $\{age{>}25, job{=}clerk\}$. A *counterfactual rule* for p is a rule of the form $p[\delta] \rightarrow y'$, for $y'{\neq}y$. We call δ a *counterfactual*. A counterfactual δ describes *what* features to change and *how* to change them to get an outcome different from y. Continuing the example, changing the income feature of x to any value >900 it will change the predicted outcome from $deny$ to $grant$. An expected property of a consistent counterfactual rule $p[\delta] \rightarrow y'$ is that it should be *minimal* with respect to x. Minimality is measured with respect to the number of split conditions sc in $p[\delta]$ not satisfied by x. We define $nf(p[\delta], x){=}|\{sc{\in}p[\delta] | \neg sc(x)\}|$, where $nf(\cdot, \cdot)$ stands for number of falsified split conditions. For example, $\{income{>}900\}{\rightarrow}grant$ is a minimal counterfactual with one condition falsified by x. In summary, a counterfactual δ is a (minimal) *motivation for reversing* the decision outcome.

B. Neighborhood Generation

The first step of LORE to extract an explanation e is the *neighborhood generation* aiming to identify a set of instances Z, with feature characteristics close to the ones of x, that is able to reproduce the local decision behavior of the black-box b. Since the objective is to learn a predictor, the neighborhood should be flexible enough to include instances with both decision values, namely $Z = Z_{=} \cup Z_{\neq}$ where instances $z \in Z_{=}$ are such that $b(z) = b(x)$, and instances $z \in Z_{\neq}$ are such that $b(z) \neq b(x)$. We extract balanced subsets $Z_{=}$ and $Z_{\neq}$, and then put $Z = Z_{=} \cup Z_{\neq}$. This task differs from approaches to instance *selection* based on genetic algorithms [7]. In our case, we cannot assume the availability of the training set of b, or not even that b is a supervised ML predictor for which a training set exists. Our task is instead similar to instance *generation* in the field of active learning, including evolutionary approaches [8]. We adopt an approach based

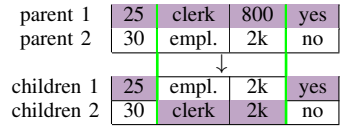

Fig. 1. Crossover.

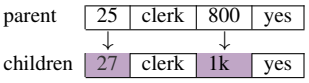

Fig. 2. Mutation.

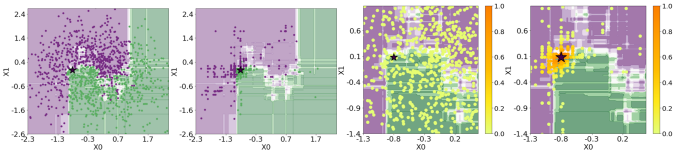

Fig. 3. Black box decision: purple vs green. Starred instance x. Uniformly random (1^{st}) and genetic generation (2^{nd}). Density of random (3^{rd}) and genetic generation (4^{th}). The colour bar in the last two plots indicates the density level (best view in color).

on a *genetic algorithm* which generates $z \in Z_{=} \cup Z_{\neq}$ by maximizing the following fitness functions:

$$fitness^{x}_{=}(z) = I_{b(x)=b(z)} + (1 - d(x,z)) - I_{x=z}$$
$$fitness^{x}_{\neq}(z) = I_{b(x)\neq b(z)} + (1 - d(x,z)) - I_{x=z}$$

where $d : \mathcal{X}^{(m)} \rightarrow [0,1]$ is a distance function, I_{true}=1, and I_{false}=0. The first fitness function looks for instances z similar to x (term $1-d(x,z)$), but not equal to x (term $I_{x=z}$) for which b produces the same outcome as x (term $I_{b(x)=b(z)}$). Thus, the maximization of $fitness^{x}_{=}$ occurs for instances different from x and whose prediction is equal to $b(x)$. The second fitness function leads to the generation of instances z similar to x, but not equal to it, for which b returns a different decision.

LORE generates Z by instantiating the evolutionary approach of [9]. Using the terminology of [8], it is an instance of generational genetic algorithms for evolutionary prototype generation. However, prototypes are a condensed subset of a training set that enable optimization in predictor learning. We aim instead to generate new instances that separate well the decision boundary of the black-box b. The *neighborhood generation* function first initializes the population P_0 with N copies of the instance x to explain. Then it enters the evolution loop that begins with the *selection* A of the P_i population having the highest fitness score. After that, the crossover operator is applied to a proportion of A according to the pc probability. The resulting and the untouched individuals are placed in B. We use a *two-point crossover* which selects two parents and two crossover features at random, and then swap the crossover feature values of the parents (see Figure 1). Thereafter, a proportion of B, determined by pm, is mutated and placed in C. The unmutated individuals are also added to C. Mutation consists of replacing features values at random according to the empirical distribution of a feature (see Figure 2). In experiments, we derive such distribution from the test set of instances to explain. Individuals in $C = P_{i+1}$ are evaluated according to the fitness function, and the evolution loop continues until G generations are completed. The best individuals, according to the fitness function, are returned. The *neighborhood generation* function is run twice, once using $fitness^{x}_{=}$ to derive $Z_{=}$, and once using $fitness^{x}_{\neq}$ to derive $Z_{\neq}$.

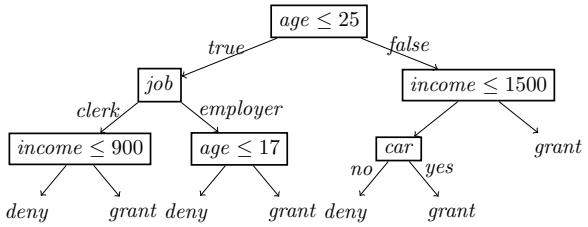

Fig. 4. Decision tree mimicking the local behavior of a black box.

Figure 3 shows an example of neighborhood generation for a black-box consisting of a random forest and a bi-dimensional feature space. The figure contrasts uniform random generation around a specific instance x (starred) to our genetic approach. The latter yields a neighborhood that is considerably denser than the former one in the boundary region of the predictor. The density of the generated instances will be a key factor in extracting a good local interpretable predictors.

C. Local Classifier and Explanation Extraction

Given the neighborhood Z of x, the second step of LORE is to build an interpretable predictor c trained on the instances $z \in Z$ labeled with the black-box decision $b(z)$ to locally mimic the behavior of b in Z. Since c must be interpretable and able to provide a factual and counter-factual explanation e, LORE considers decision tree classifiers as interpretable predictor: *(i)* decision rules can naturally be derived from a root-leaf path in a decision tree; and, *(ii)* counterfactuals can be extracted by symbolic reasoning over the tree.

Once the decision tree c has been trained on Z labeled with b, LORE derives the explanation $e = \langle r, \Phi \rangle$ as follows. The decision rule $r = p \rightarrow y$ is formed by including in p the split conditions on the path from the root to the leaf node that is satisfied by x, and setting $y = c(x)$. By construction, r is consistent with c and satisfied by x. Consider now the counterfactual rules in Φ. LORE looks for all paths in the decision tree c leading to a decision $y' \neq y$. Fix one of such paths, and let q be the conjunction of split conditions in it. Again by construction, $q \rightarrow y'$ is a counterfactual rule consistent with c. Notice that, since we are using a decision tree, the counterfactual δ for which $q = p[\delta]$ has not to be explicitly computed – this is a benefit of using decision trees. Among all such q's, only those with the minimum number of split conditions sc not satisfied by x are kept in Φ.

As an example, consider the decision tree in Figure 4, and the instance $x = \{(age{=}22), (job{=}clerk), (income{=}800), (car{=}no)\}$ for which the decision *deny* (e.g., of a loan) has to be explained. The path followed by x is the leftmost one in the tree. The *factual* decision rule extracted from the path is $\{age \leq 25, job = clerk, income \leq 900\} \rightarrow deny$. There are four paths leading to the opposite decision: $q_1 = \{age \leq 25, job = clerk, income > 900\}$, $q_2 = \{17 < age \leq 25, job = employer\}$, $q_3 = \{age > 25, income \leq 1500, car = yes\}$, and $q_4 = \{age > 25, income > 1500\}$. It turns out: $nf(q_1, x) = 1$, $nf(q_2, x) = 1$, $nf(q_3, x) = 2$, $nf(q_4, x) = 2$, and $\Phi = \{q_1 \rightarrow grant, q_2 \rightarrow grant\}$.

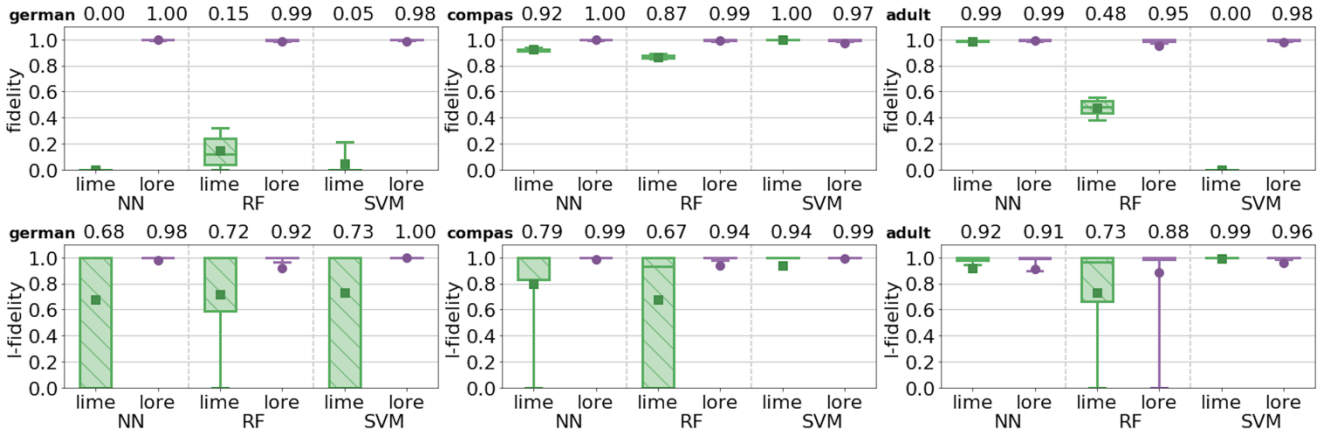

Fig. 5. LORE vs LIME: box plots of *fidelity* and *l-fidelity*. Top numbers are mean values.

A *counterfactual instance* can also be generated from x and from counterfactual rule $q \rightarrow y'$. Among all instances that satisfy q, we choose the one that minimally changes attributes from x according to q.

III. Experiments

In this section, we evaluate LORE against state-of-the-art competitors[1]. LORE has been developed in Python, using the *deap*[2] library for the genetic neighborhood generation, and the *YaDT* system[3] for the decision tree induction.

A. Experimental Setup

We ran experiments on three tabular real-world open source datasets: `adult`[4], `compas`[5] and `german`[6]. Each dataset was randomly split into two parts: 70% was used to train the black-box classifiers, 30%, denoted by X, was used as instances to be explained. We denote by $\hat{Y}$ the decisions provided by the black-box b and by Y the set of decisions provided by the interpretable predictor c. We trained and explained away the following black-box classifiers: Random Forest (RF), Support Vector Machine (SVM), and multi-layer perceptron (NN) as implemented by the *scikit-learn* Python library. Default parameters were used for both the black-boxes and the libraries of LORE[7]. We consider the following performance indicators to evaluate the quality of the explanations:

- The ***fidelity***$(Y, \hat{Y}) \in [0, 1]$ compares the predictions of c and b on Z measuring how good is c at mimicking b.
- The ***l-fidelity***$(Y, \hat{Y}) \in [0, 1]$ compares the predictions of c and b on the local (hence "l-") instances of Z covered by r measuring how good is r at mimicking b.
- The ***hit***$(y, \hat{y}) \in \{0, 1\}$ compares the predictions of c and b on the instance x under analysis. It returns 1 if $c(x)$ is equal to $b(x)$, and 0 otherwise.

We adopt the F1-measure for the first two. Aggregated values are reported by averaging performance indicators over X.

TABLE I
LORE vs LIME: *hit* scores.

Dataset	`german`		`compas`		`adult`	
Black Box	LORE	LIME	LORE	LIME	LORE	LIME
RF	**.925 ± .2**	.880 ± .3	**.941 ± .2**	.826 ± .4	**.901 ± .3**	.824 ± .4
NN	.980 ± .1	**1.00 ± .0**	**.987 ± .1**	.902 ± .3	.918 ± .3	**.998 ± .1**
SVM	**1.00 ± .0**	.966 ± .1	**.997 ± .1**	.900 ± .3	.985 ± .1	**.987 ± .1**

B. Rules vs Linear Regression

We present a qualitative and quantitative comparison with the linear explanations of LIME[8] [3]. A first crucial difference is that in LIME, the number of features composing an explanation is an input parameter that must be specified by the user. LORE, instead, automatically provides the user with an explanation including only the features useful to justify the black-box decision. This is a clear improvement over LIME. In experiments, unless otherwise stated, we vary the number of features of LIME explanations from two to ten, and we consider the performance with the highest hit score.

Table I reports the mean and standard deviation of *hit*, while Figure 5 details the box plots of *fidelity* (top) and *l-fidelity* (bottom). The results show that LORE definitely outperforms LIME under various viewpoints. Regarding the *hit* score, LORE is clearly better than LIME in 6 out of 9 cases, is very close to it in 2 cases, and performs clearly worse in 1 case. LORE has better *fidelity* scores and is more robust than LIME, which, instead, exhibits very high variability in the neighborhood (i.e., *l-fidelity*). This result can be attributed to the genetic approach of LORE. Figure 7 reports a multidimensional scaling of the neighborhoods of an instance x generated by the two approaches. LORE computes a dense and compact neighborhood. The instances generated by LIME, instead, can be very distant from each other and with low density around x.

[1]The source code and the datasets for reproducing the experiments are publicly available at https://github.com/riccotti/LORE. Experiments were performed on Ubuntu 16.04 LTS, 32 GB RAM, 3.30GHz Intel Core i7.

[2]https://github.com/DEAP/deap

[3]http://pages.di.unipi.it/ruggieri/YaDT/

[4]https://archive.ics.uci.edu/ml/datasets/adult

[5]https://github.com/propublica/compas-analysis

[6]https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)

[7]The parameters of the genetic procedure, namely probabilities of crossover and mutation, number of generations, and population size are set with the default values [9] of 0.7, 0.2, 10 and 1000, respectively.

[8]https://github.com/marcotcr/lime

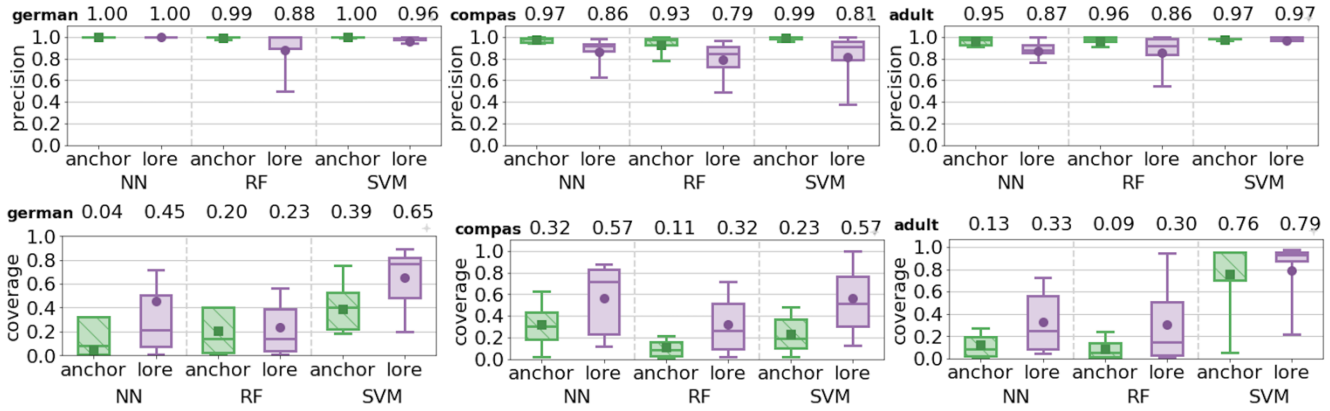

Fig. 6. LORE vs ANCHOR: box plots of *precision* and *coverage*. Top numbers are mean values.

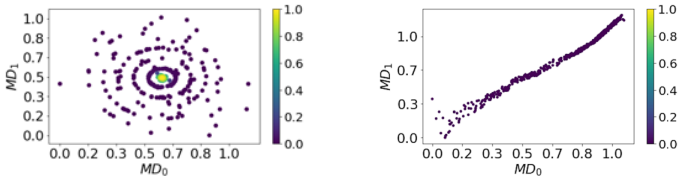

Fig. 7. Neighborhoods multidimensional scaling of an instance x. LORE (left) computes a dense and compact neighborhood. LIME (right) generates instances that can be very distant from each other and with low density around x.

We claim that the explanations provided by LORE are more abstract and comprehensible than the ones of LIME. Consider the example in Figure 8. The top part reports a LORE local explanation for an instance x from `german`. The central part is a LIME explanation. Weights in w are associated with the categorical values in the instance x to explain, and to continuous upper/lower bounds where the bounding values are taken from x. Each weight tells the user how much the decision would have changed for different values of a specific feature. In the example, the weight 0.11 has the following meaning [3]: "if the duration in months had been higher than the value it is for x, the prediction would have been, on average, 0.11 less *0* (or 0.11 more *1*)"[9]. A not very easy logic to follow when compared to a single decision rule which characterizes the contextual conditions for the decision of the black-box. Another advantage of LORE explanation consists in the set of counterfactual rules. LIME provides a rough indication of where to look for a different decision: different categorical values or lower/higher continuous values of some feature. LORE's counterfactual rules provide high-level and minimal-change contexts for reversing the outcome prediction.

C. Rules vs Anchors

ANCHOR [4] is a rule-based local explainer inspired to LIME, which provides decision rules, called *anchors*[10], as explanations. Anchors are computed by incrementally adding equality conditions in the premise, while an estimate of the anchor precision is above a minimum threshold (set to 95%).

[9]See *lime-tutorial* for more details about LIME explanations
[10]https://github.com/marcotcr/anchor

- LORE
  $r =$ *({credit_amount > 836, other_debtors = none, credit_history = critical account} → decision = 0)*
  $\Phi =$ *{ ({credit_amount ≤ 836, other_debtors = none, credit_history = critical account} → decision = 1), ({credit_amount > 836, other_debtors = none, credit_history = all paid back} → decision = 1) }*
- LIME
  $w =$ *[({duration_month ≤ 12:0.11, status=male_single: 0.07, credit_history= critical account: 0.06}, decision = 0), ({account_check_status = 0 ≤ ... < 200 DM: 0.09, installment_as_income= 1: 0.07}, decision = 1)]*
- ANCHOR
  $a =$ *({credit_history = critical account, duration_month ∈ [0, 18.00]} → decision = 0)*

Fig. 8. Examples of explanations from LORE, LIME and ANCHOR.

TABLE II
LORE VS ANCHOR: JACCARD MEASURE OF STABILITY.

Dataset	german		compas		adult	
Black box	LORE	ANCHOR	LORE	ANCHOR	LORE	ANCHOR
RF	**.76** ± **.15**	.61 ± .15	**.75** ± **.12**	.73 ± .14	**.70** ± **.15**	.69 ± .15
NN	**.69** ± **.18**	.53 ± .21	**.83** ± **.13**	.79 ± .16	**.81** ± **.12**	.65 ± .16
SVM	**.82** ± **.16**	.32 ± .16	**.71** ± **.16**	.70 ± .20	**.87** ± **.14**	.67 ± .13

Such an estimation relies on neighborhood generation through pure-exploration multi-armed bandit. On a qualitative level of comparison, ANCHOR requires the *apriori* discretization of continuous features, while LORE benefits of the capabilities of decision trees to split continuous features (see Figure 8).

On a quantitative level of comparison, since Anchor produces a single rule model starting from the instance to explain, hit is 100% by construction. Moreover, l-fidelity boils down to rule precision, namely the fraction of instances in the neighborhood set that is correctly classified by the rule, i.e., that have the same black-box prediction as the instance to explain. Figure 6 reports the average precision of the decision rules for both ANCHOR and LORE. By construction, rule precision in ANCHOR is very high, since an estimator of such precision is constrained to be at least 95%. We also evaluate the rule coverage, namely the fraction of instances to explain covered by the rule: large values of coverage means better rule generalization [4]. Figure 6 also shows the average coverage

TABLE III
LORE VS SOC: PERFORMANCE OF COUNTERFACTUAL RULES.

dataset	*method*	*nf*	*c-hit*	*cl-fidelity*
german	LORE	**1.52 ± 1.18**	**.7765 ± .38**	**.6355 ± .43**
	SOC	14.80 ± 1.59	.3118 ± .47	.2297 ± .36
compas	LORE	**1.84 ± 0.78**	**.8694 ± .37**	**.8611 ± .41**
	SOC	6.24 ± 1.45	.8036 ± .38	.7555 ± .34

of the decision rules, with LORE showing a consistently better coverage than ANCHOR. We also measure the possible over-specialization of the decision rules by evaluating their stability with respect to randomness in the neighborhood generation. We measure stability using Jaccard coefficient of feature sets used in the 10 decision rules computed for the same instances in 10 runs of the system. Table II reports mean and standard deviation of Jaccard coefficient. LORE has a better stability than ANCHOR for all datasets and black-boxes. In summary, ANCHOR shows better precision than LORE at the expenses of generality and stability of the produced explanations. Such two properties are, however, essential for a general acceptance of an explanation methodology.

D. Rule-Based vs Stochastic Counterfactuals

We compare LORE with the stochastic optimization counterfactual (SOC) approach [10] returning an instance x' as close as possible to a given x, but for which the black-box outputs a different prediction. To make a fair comparison, we have implemented the SOC as an alternative fitness function of the genetic neighborhood generation. Table III shows the performances of the two methods on nf (number of falsified conditions in counterfactual rules), c-hit (rate of agreement of black-box and counterfactual decision for counterfactual instance), and cl-$fidelity$ (F1-score of agreement of black-box and counterfactual decision). Results show that LORE returns shorter explanations, i.e., simpler explanations, and provides counterfactual rules with an higher fidelity than SOC.

IV. Future Works

Future research directions include multi-valued classification, going beyond relation data towards image and text, going beyond decision trees, and considering alternative models such as rule sets and rule lists. Moreover, even though LORE should be able to deal with high dimensional datasets via the genetic process, we plan to develop further experiments to verify its robustness. We are fully aware that the quest towards meaningful explanations of black-box systems is at an embryonic stage. How to turn our proposed factual and counterfactual rules into substantive narratives, which can empower human stakeholders with diverse expertise by boosting their causal and what-if reasoning, is a fascinating challenge of great practical relevance for the successful adoption of many AI innovations.

Acknowledgment

This work is partially supported by the European Commission through the H2020 project INFRAIA-1-2014-2015: Research Infrastructure G.A. 654024 *SoBigData*, G.A. 825619 *AI4EU*, G.A. 761758 *Humane AI*, and the ERC-2018-ADG G.A. 834756 "XAI: Science and technology for the eXplanation of AI decision making".

References

[1] R. Guidotti *et al.*, "A survey of methods for explaining black box models," *ACM Computing Surveys*, vol. 51, no. 5, p. 93, 2018.
[2] S. Grover, C. Pulice, G. I. Simari, and V. Subrahmanian, "Beef: Balanced english explanations of forecasts," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 2, pp. 350–364, 2019.
[3] M. T. Ribeiro *et al.*, ""Why should I trust you?": Explaining the predictions of any classifier," in *KDD*. ACM, 2016, pp. 1135–1144.
[4] ——, "Anchors: High-precision model-agnostic explanations," in *AAAI*. AAAI Press, 2018, pp. 1527–1535.
[5] B. Ruth M.J., "Counterfactuals in explainable artificial intelligence (xai): Evidence from human reasoning," *IJCAI*, pp. 6276–6282, 2019.
[6] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artificial Intelligence*, vol. 267, pp. 1 – 38, 2019.
[7] J. A. Olvera-López *et al.*, "A review of instance selection methods," *AIR*, vol. 34, no. 2, pp. 133–143, 2010.
[8] J. Derrac *et al.*, "A survey on evolutionary instance selection and generation," in *Modeling, Analysis, and Applications in Metaheuristic Computing*. IGI Global, 2012, pp. 233–266.
[9] T. Bäck, D. B. Fogel, and Z. Michalewicz, *Evolutionary computation 1: Basic algorithms and operators*. CRC press, 2000, vol. 1.
[10] S. Wachter *et al.*, "Counterfactual explanations without opening the black box: Automated decisions and the GDPR," *HJLT*, 2018.

Riccardo Guidotti is a researcher at KDD Lab in ISTI-CNR, Pisa. He holds a PhD in Computer Science from the University of Pisa, with a thesis on Personal Data Analytics. He received an IBM fellowship Award in 2014/2015, and spent an Internship at IBM Research Dublin. He has been awarded with the ISTI-CNR Young Researcher Award from 2016 to 2019, and with the NGDSA at DSAA 2018.

Anna Monreale is an Assistant Professor at the Computer Science Department, University of Pisa and member of KDD Lab. She has been a visiting student at Stevens Institute of Technology in 2010. Her research interests include data analytics, privacy and ethical issues in mining human data. She is a Privacy-by-Design Ambassador and a member of the EU Panel of Experts on the Open Science Cloud.

Fosca Giannotti is a director of research at ISTI-CNR, Pisa. Fosca leads the Pisa KDD Lab - Knowledge Discovery and Data Mining Laboratory, a joint research initiative of ISTI-CNR and University of Pisa, founded in 1994. She is the coordinator of the European research infrastructure SoBigData (http://www.sobigdata.eu). She received the ERC Advanced Grant XAI – *Science and technology for the explanation of AI decision making.*

Dino Pedreschi is a professor of Computer Science at the University of Pisa, a co-founder of KDD Lab, and a pioneering scientist in data science. His research focus is on big data analytics and mining and their impact on society. Dino is the director of the Data Science PhD program at Scuola Normale Superiore in Pisa.

Salvatore Ruggieri is professor of Computer Science at the University of Pisa. He is member of KDD Lab, with research interests in data mining and AI, including: algorithmic fairness, explainable AI, systems for knowledge discovery process; sequential and parallel classification algorithms. He is a program co-chair of the ACM conference on Fairness, Accountability and Transparency, 2020.

Franco Turini is a professor in the Department of Computer Science, University of Pisa and a co-founder of KDD Lab. In 78/80 he has been a visiting scientist of the Carnegie-Mellon University and of the IBM Research Center S.Jose, afterwards. In 92/93 he has been visiting professor at the University of Utah. His contributions include discrimination-aware data mining and ML.