Artificial Intelligence 291 (2021) 103428

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Evaluating local explanation methods on ground truth ☆

Riccardo Guidotti [a,b,*]

[a] University of Pisa, Largo B. Pontecorvo, Pisa, Italy
[b] ISTI-CNR, Via G. Moruzzi, Pisa, Italy

ARTICLE INFO

Article history:
Received 22 November 2019
Received in revised form 26 October 2020
Accepted 7 November 2020
Available online 14 November 2020

Keywords:
Evaluating explanations
Explainable AI
Interpretable models
Open the black box
Local explanation

ABSTRACT

Evaluating local explanation methods is a difficult task due to the lack of a shared and universally accepted definition of explanation. In the literature, one of the most common ways to assess the performance of an explanation method is to measure the fidelity of the explanation with respect to the classification of a black box model adopted by an Artificial Intelligent system for making a decision. However, this kind of evaluation only measures the degree of adherence of the local explainer in reproducing the behavior of the black box classifier with respect to the final decision. Therefore, the explanation provided by the local explainer could be different in the content even though it leads to the same decision of the AI system. In this paper, we propose an approach that allows to measure to which extent the explanations returned by local explanation methods are correct with respect to a synthetic ground truth explanation. Indeed, the proposed methodology enables the generation of synthetic transparent classifiers for which the reason for the decision taken, i.e., a synthetic ground truth explanation, is available by design. Experimental results show how the proposed approach allows to easily evaluate local explanations on the ground truth and to characterize the quality of local explanation methods.

1. Introduction

In the past decade, we have been witnessing the increasing deployment of powerful automated Artificial Intelligence's (AI) decision-making systems in settings ranging from pedestrian detection on self-driving cars to evaluation of loan concession in bank systems. While evidently powerful in solving complex tasks, these systems are typically opaque, i.e., they provide hardly any mechanisms to explore and understand their behavior and the reasons underlying the decisions taken. The ubiquitous usage of opaque machine learning algorithms in AI systems is leading to the rise of a *black box society* [1,2]. The benefits of extremely accurate systems come with the price of their opaqueness that on its own leverages numerous legal, ethical and practical issues. Indeed, this is problematic not only for lack of transparency but also for possible biases inherited by the AI from collection of artifacts and human prejudices hidden in the training data, which may lead to unfair or wrong decisions [3]. Well-known examples are the Compas[1] and Amazon[2] cases where the AI black box models were taking decisions discriminating ethnic minorities. To prevent such cases, the EU has introduced the *General Data Protection*

☆ This article belongs to Special Issue: Explainable AI.
* Corresponding author at: University of Pisa, Largo B. Pontecorvo, Pisa, Italy.
E-mail address: riccardo.guidotti@unipi.it.
[1] http://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.
[2] http://www.businessinsider.com/how-algorithms-can-be-racist-2016-4.

https://doi.org/10.1016/j.artint.2020.103428

Regulation[3] which gives to individuals the right to request "...meaningful information of the logic involved" when automated decision-making takes place with "legal effects" on individuals [4–6].

As a consequence, nowadays there is high attention in research on methods for explaining AI black box classifiers [7–9,3,10]. These explanation methods are distinguished as *model-agnostic* or *model-specific* if can explain the decision of any classifier or just of a particular one, and as *local* or *global* if can explain the reasons for a single decision, or the overall logic of a classifier [3]. In particular, due to their recent growth in the literature, in this paper, we focus on local model-agnostic explanation methods, and because of the crucial and difficult nature of the problem, we point the attention on the *evaluation* of the explanation extracted by these methods.

The most used evaluation measure for local explanation methods is the so-called *fidelity*, i.e., the degree of approximation with respect to the AI black box that the interpretable surrogate model learned in the neighborhood of the instance to explain is able to achieve [3,11]. The fidelity is generally measured as the accuracy of the interpretable model with respect to the black box. However, this kind of evaluation does not take into account the *content* of the explanation, but only compares the decision of the black box with the decision provided according to the explanation. This is due to the fact that the *ground truth* for explanations is not available. On the other hand, if one is aware of which are the real reasons for which an AI system takes individual decisions, i.e., the ground truth, then it is possible to judge to which extent the explanations returned by an explanation method are correct. In this paper, we indicate with the term ground truth explanation the reasons for which the black box returned a certain outcome for a specific instance. It is fundamental to highlight that the explanation provided by a local explainer could be not correct, i.e., not similar to the ground truth explanation, even though it leads to the same decision of the AI system. For instance, if the ground truth explanation for a loan request is $\textit{if } age \leq 25 \wedge income > 1500 \textit{ then deny}$, and the retrieved explanation is $\textit{if } age \leq 20 \wedge children > 1 \textit{ then deny}$, then the outcome is the same, while the reasons are only partially in agreement, i.e., the retrieved explanation is not entirely *correct*. Paraphrasing Seneca: "Not only how faithful, but also how well you explain is the main thing". Therefore, the objective of this work is to propose a way for *quantitatively* estimating the correctness of local explanations.

In the literature it is recognized that *synthetic data* has certain advantages over real-world data [12] for evaluating trained machine learning models adopted by AI systems. Thus, *synthetic classifiers* generated with a controlled approach can have advantages in evaluating local explanation methods because *synthetic ground truth explanations* available from synthetic *transparent* classifiers can be exploited for evaluating explanation methods. To the best of our knowledge, there are no works solving the problem of evaluating a local model-agnostic explainer with synthetic ground truth explanations. Therefore, we overcome existing limitations with the **S**ynthetic **E**xplai**N**abl**E** Cl**A**ssifier (SENECA) generators. SENECA is an array of methods for *generating synthetic transparent classifiers* for which the reasons for individual decisions, i.e., the *synthetic ground truth explanations, are available by design. The generated synthetic transparent classifiers allow to measure quantitatively the correctness of the explanations returned by local explanation methods with respect to synthetic ground truth explanations* for tabular data, images and text. SENECA allows to empirically measure to which extent the mimic of a local explanation model follows the same logic of the machine learning model explained.

Experimental results show how, thanks to SENECA generators, it is possible to pursue a systematic evaluation and a statistically significant comparison of a broad array of local model-agnostic explainers. A massive number of synthetic explainable classifiers for tabular data, images, and text with different characteristics are generated in a controlled way such that it is possible to stress out the explainers and understand their strengths and weaknesses.

The practicality of the proposed approach is twofold. First, to the best of our knowledge, it is the only proposal for evaluating the explanations based on the content of the explanation itself, instead of based on the outcome they lead to. The evaluation measures made available by SENECA can be considered as an "improved fidelity", not considering only the outcome but the whole explanation. Second, since it does not exist a standardized way to benchmark local explanation methods in terms of the content of the explanation, by exploiting SENECA would be possible to compare the performance of local explanation methods with respect to multiple and different transparent classifiers that can be "similar" to the real and not interpretable one a user could be interested in explaining. This can lead to selecting the best local explanation method. However, it is worth to underline that, a domain-specific human-based evaluation would be obviously recommended for an explanation method deployed in a real setting because, since SENECA evaluations are calculated on synthetic classifiers working on synthetic datasets cannot account for domain-specific aspects. In addition, due to the growing proposal of local explanation methods, SENECA can be used for benchmarking and testing novel local explainers compared to existing ones.

The rest of the paper is organized as follows. In Section 2 we review work proposing the generation of synthetic datasets in various problem settings. Section 3 recalls basic notions related with local model-agnostic explainers. In Sections 4 and 5 we illustrate the proposed method and evaluation measures, respectively. Section 6 shows a wide experimentation stressing with SENECA state-of-the-art local model-agnostic explainers. Finally, Section 7 summarizes the contribution and proposes future research directions.

[3] https://ec.europa.eu/justice/smedataprotect/.

2. Related work

Although the explanation of machine learning models for AI systems is largely studied and debated, there are no existing works evaluating local model-agnostic explainers by means of synthetic ground truth explanations.

However, we can observe in the literature a parallel between the issues analyzed in checking the degree of correctness of an explanation, and those faced in testing accuracy in unsupervised data mining algorithms. In both cases, the main problem is that the *ground truth* is missing. With respect to unsupervised algorithms, the solution generally adopted is the generation of *synthetic* datasets with a ground truth enclosed. In such a way, the analyzed algorithms can be tested and stressed by varying the parameters of the synthetic dataset generation. On the other hand, in our case we aim at generating synthetic transparent classifiers (not dataset) to retrieve synthetic explanation to be used as ground truth.

In [13] is presented an approach for *general-purpose clustering* based on distribution and transformation to generate synthetic data for multi-dimensional numerical datasets. Similarly, in [14] are proposed two synthetic generative models to stress clustering algorithms for the task of location detection in mobility data mining: a generator acting at random, and a generator following users' mobility rules inferred from real data. Moving to *clustering of transactional data*, in [15] it is designed a synthetic transaction generation process parametric to the number of transactions, items, average transaction size, clusters, outliers, and degree of overlap among transactions. It is exploited also in [16,17] to compare novel algorithms. The generation of synthetic ground truth is largely investigated in the field of *community detection*. In [18] it is proposed an approach to generate directed and weighted networks with built-in community structure, also considering the possibility that nodes belong to more communities. In [19], structural communities of complex networks are opposed to ground truth to find the metadata groups in large networks. In [20] it is presented a methodology for comparing and quantitatively evaluate how different structural definitions of communities correspond to ground-truth functional communities. In [21] it is designed a faster alternative to [18] adopting a different quality measure for the evaluation during the generation.

Moreover, since there are some disadvantages in using real-world data (e.g., hand collection, small size, etc.), there are various other fields of research exploiting synthetic data generation to analyze algorithms performance. In [12] it is proposed a data generator for the evaluation of supervised and unsupervised methods that produces synthetic data with guaranteed global and class-specific statistical properties. In [22] it is described a generator of synthetic gene expression for the analysis of structure learning algorithms. In particular, it creates synthetic transcriptional regulatory networks and produces simulated gene expression data that approximate experimental data. Finally, synthetic data generation can be useful for database and software testing. In [23] it is developed an extrapolation system that generates a representative database given a structure and a scaling rate.

Explanation methods are evaluated with ground truth explanations in [24–26]. However, the synthetic classifiers adopted in these papers are extremely simple, not customizable nor tunable, i.e., the synthetic classifiers are ad-hoc functions or rules which are hand-defined and not generated. Therefore, there is a crucial difference between the work presented in this paper, and the synthetic classifiers adopted in [24–26]. Indeed, while in [24–26], are adopted a priori synthetic classifiers, in this paper we propose an approach to generate synthetic transparent classifiers.

In line with the motivations of the works presented above, and in order to spread the usage of synthetic classifier generators, in this paper we propose synthetic transparent classifier generators for testing and comparing local model-agnostic explanation methods.

3. Setting the stage

In order to understand the motivations and the design of SENECA, in the following, we recall the problem definition that it contributes to analyzing.

A classifier is a function $f : \mathcal{X}^{(m)} \rightarrow \mathcal{Y}$ mapping instances (tuples) x from a feature space $\mathcal{X}^{(m)}$ with m features to a decision y in a target space $\mathcal{Y}$ of size $l = |\mathcal{Y}|$, i.e., y can assume l different labels.[4] We write $f(x) = y$ to denote the decision y of f, and $f(X) = Y$ as a shorthand for $\{f(x) \mid x \in X\} = Y$. An instance x consists of a set of m attribute-value pairs (a_i, v_i), where a_i is a feature[5] (or attribute) and v_i is a value from the domain of a_i.

We can distinguish between opaque and transparent classification functions [11]. In particular, we denote with b a *black box* classifier used by an AI system, whose internals for returning the decision $b(x)$ are either unknown to the observer or they are known but uninterpretable by humans. Examples include neural networks, SVMs, ensemble classifiers, etc [27]. On the other hand, we denote with c an *interpretable* (comprehensible) classifier, whose internal processing yielding a decision $c(x)$ can be given an interpretation understandable by a human. Examples include rule-based classifiers, decision trees, decision sets, and linear models [27].

An important problem that is faced nowadays in the literature is the *black box outcome explanation problem* [3,8]. Given a black box b and an instance x, the *black box outcome explanation problem* consists in providing an explanation $e \in E$ belonging to a human-interpretable domain E for the decision $b(x) = y$. The problem is generally addressed by learning

[4] If $l = 2$ we are dealing with a binary problem, with $l > 2$ with a multiclass problem. Without losing in generality, in this paper we assume $l = 2$.

[5] The domain of a feature can be continuous or categorical. A categorical feature can be turned into a set of continuous features using techniques like one-hot encoding [27]. Without loosing in generality, in the following we consider only continuous features.

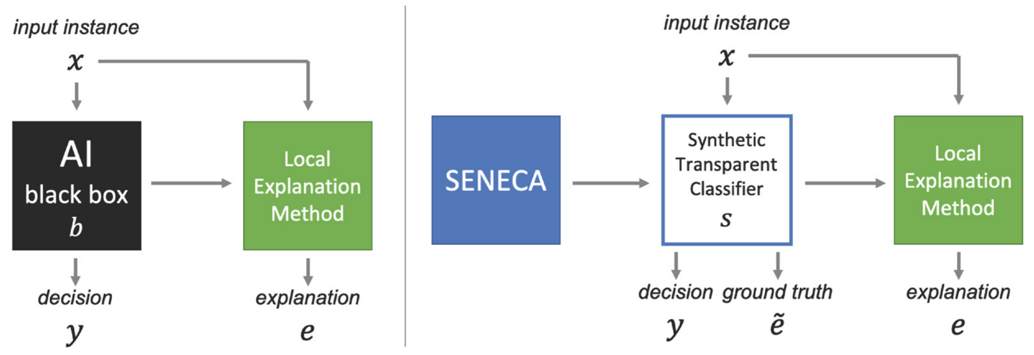

Fig. 1. *Left:* Local explanation method that takes as input an instance x an AI using a black box function b and returns the explanation e. *Right:* The SENECA generator returns a synthetic transparent classification function s such that the explanation e returned by the local explanation method can be compared with the ground truth explanation $\tilde{e}$.

an interpretable surrogate model c that reproduces and accurately mimes the *local* behavior of b. An explanation e of the decision is then derived from c. Fig. 1 (left) illustrates this explanation process.

Given an instance x for which an explanation is required for the decision $y = b(x)$, in the literature [3] we can distinguish two forms of *human-interpretable domains* E and consequently two forms of local explanations: *(i)* feature importance explanations, and *(ii)* rule-based explanations.

A *feature importance explanation* is modeled as a vector $v = \{v_1, v_2, \dots, v_m\}$ where the value $v_i \in v$ is the importance of the i^{th} feature [3,28,29]. Every value v_i is informative in two ways: the sign and the magnitude. If $v_i > 0$, it means that feature contributes positively to the get the outcome y otherwise, it contributes negatively. With respect to the magnitude, the greater is the value of $|v_i|$, the greater its contribution. A value of $v_i = 0.0$ means no contribution from the i^{th} feature for the decision of x. The most well-known model-agnostic local explanation methods returning features importance as explanation are LIME [28], MAPLE [30] and SHAP [31]. An example of a feature based explanation is $e = \{age = 0.8, income = 0.0, children = -0.2\}$, $y = deny$, meaning that *age* is the most important feature for the decision *deny* for a load request with a positive contribution, *children* has a small negative contribution, while *income* is not affecting the outcome. For images and texts, the features correspond to pixels, or words, respectively [28].

LIME [28] randomly generates a local neighborhood around the instance x for which an explanation is required. Then, it trains a linear model c (a Lasso regression classifier) on the neighborhood annotated with the AI black box b and returns the feature importance vector v as an explanation. MAPLE [30] exploits a random forest for the selection of the features used by a local linear model c (as a Ridge regression classifier) in the randomly generated neighborhood of x. SHAP [31] tries to overcomes LIME limitations' exploiting the *Shapely* values of a conditional expectation function of the black box by providing a unique additive feature importance. Another proposal is presented in [25]: the explainer fits a local subspace neighborhood around the instance to explain using the local intrinsic dimensionality, then generates neighbors in the local subspace and projects them back to the original space. The strength of these approaches lies in the fact that, besides tabular data, they can be employed to explain black box classifiers of AI systems working on text and images [3]. In that cases, the features importance turns generally into *word importance* and to *pixel importance* (e.g., in forms of saliency maps [29,32–34]), for text and images respectively.

Besides features importance based explainers, in the literature, we can find rule-based explainers. A *rule based explanation* is a decision rule r of the form $r = p \to y$, the decision y is the *consequence* of the rule, while the *premise* p is a boolean condition on feature values [3,35,36]. In particular, 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 $\pm\infty$.[6] Given r as explanation for x, means that x *satisfies* r, or r *covers* x, i.e., the boolean condition p is evaluated true for x. An example of a rule-based explanation is $e = \{\{age \leq 25, income > 1500\} \to deny\}$.

Well-known rule-based model-agnostic local explanation methods are ANCHOR [36], BRL [37], and LORE [35]. ANCHOR [36] uses a bandit algorithm that randomly constructs the "anchor rules" with the highest coverage and respecting a precision threshold. LORE [35] adopts a genetic procedure to generate the local neighborhood and learns decision and counterfactual rules from that neighborhood using a decision tree as local interpretable classifier c. The BRL method [37] given a data distribution, extracts a sample exploiting the distribution containing the instance to explain. Then it labels this set using the black box and trains on them a Bayesian rule list c used to explain the instance. Rule-based local explainers are less used than features importance explainers because they are not effective for images and text. However, for tabular data, an explanation given as a rule r is generally much more interpretable than an explanation given as a vector v of numbers [11].

[6] Using $\pm\infty$ we can model with a single notation typical univariate split conditions, such as equality ($a = v$ as $a \in [v, v]$), upper bounds ($a \leq v$ as $a \in [-\infty, v]$), strict lower bounds ($a > v$ as $a \in [v + \epsilon, \infty]$ for a sufficiently small ϵ).

4. Synthetic explainable classifier generators

In this section we present the **S**ynthetic **E**xplai**N**abl**E** **C**l**A**ssifier (SENECA) generators, an array of methods for generating synthetic transparent classifiers for tabular data, images, and text for which the explanations of individual decisions are available by design. Fig. 1 (right) shows that, thanks to a SENECA generator, it is possible to assess quantitatively the explanation power of a local explanation method returning an explanation e because, for a given instance x, the generated synthetic transparent classifier s makes available a ground truth explanation $\widetilde{e}$. A SENECA generator has no input and produces in output a synthetic transparent classifier s. Such a transparent classifier takes as input an instance x and returns the outcome y (decision) for x together with the explanation $\widetilde{e}$ for the outcome y. A local explanation method takes as input an instance x, a black box model b, and retrieves the reasons for the outcome y. SENECA allows to estimate to which extent the retrieved explanation e is similar to the ground truth explanation $\widetilde{e}$. We underline that the nature of the explanations, both the ground truth $\widetilde{e}$ and the one retrieved by the explanation method e, can vary depending on the type of data under analysis, and by the type of classifier. For instance, if the data is tabular (i.e., relational) and the classifier is rule-based, then the nature of the explanations $\widetilde{e}$ and e is that of logical rules with conditions on the features characterizing the data. As another example, if we are dealing with text and the classification is based on the positive and negative sentiment of the words expressed as weights, then the nature of the explanations $\widetilde{e}$ and e can be a vector containing the weights for the words responsible for the classification. Therefore, with the ground truth explanation $\widetilde{e}$ revealing the real reasons for the classification of an instance, it is possible to judge to which extent the explanation e is "correct". In the most naive setting e is *correct* if e is equal to $\widetilde{e}$, otherwise it is not correct. However, since retrieving the exact explanation is quite challenging and the current state-of-the-art is rarely able to achieve such a perfect match, in this paper we define an array of evaluation measures between two explanations with a co-domain of "correctness" in the range $[0, 1]$. In summary, the degree of correctness of an explanation e indicates how much it matches the ground truth explanation $\widetilde{e}$, i.e., the real reasons for which a classifier takes a decision y. SENECA allows generating synthetic classifiers s from which is possible to extract by design the ground truth explanations that enable this measurement.

In line with the state-of-the-art in terms of local model-agnostic explainers, we develop four different generators depending on the nature of the explanations, i.e., type of data and on the type of explanation:

- SENECA-RB: a generator of synthetic rule-based classifiers on tabular data returning *rules* as synthetic ground truth explanations,
- SENECA-RC: a generator of synthetic regression model on tabular data returning *features importance* as synthetic ground truth explanations,
- SENECA-IMG: a generator of synthetic image classifiers returning *pixels importance* as synthetic ground truth explanations,
- SENECA-TXT: a generator of synthetic text classifiers returning *words importance* as synthetic ground truth explanations.

We highlight that SENECA is *not* a synthetic dataset generator, and is *not* an explanation method. SENECA is a generator of synthetic transparent classifiers from which it is possible to extract local explanations. These explanations can be used as ground truth for evaluating local explanation methods with respect to the explanation's content. To this aim, SENECA generators do exploits and produce synthetic data. However, we stress the point that SENECA *generators is not a family of synthetic data generator*, and it is not the generated data that is used as ground truth but the explanations extracted from the synthetic transparent classifiers. Furthermore, we underline that the four SENECA generators allow experimenting with the vast majority of local explainers as they cover almost all the existing types of explanations and data addressed by the state-of-the-art [3]. There are no restrictions in applying SENECA for any type of dataset in the form of tabular data, images, and text. We leave as future research the development of SENECA generators for explainers on black box classifiers working on sequences, time series, and graphs.

In the following, we describe the details of each generator.

4.1. Rule-based classifier generator

The purpose of the SENECA-RB generator is to produce a rule-based classifier s from which, given an instance x, it is possible to extract the rule r responsible for the classification of x by s. As rule-based classifier s, we adopt a decision tree classifier [39] since it is simple, effective and guarantees that any instance x is covered only by one rule. The method SENECA-RB, described in Algorithm 1, starts from the generation of a binary dataset of m features (line 1) by allocating for each class one or more normally-distributed clusters of points using the procedure[7] described in [38] implemented by *makeClassificationData*. It can be summarized as follows:

1. Each of the two classes is composed by a set of Gaussian clusters. N(0,1) is used to draw for each cluster n examples of independent features.

[7] http://clopinet.com/isabelle/Projects/NIPS2003/Slides/NIPS2003-Datasets.pdf.

Algorithm 1: SENECA-RB(m, α, β).

```
Input  : m - number of features, α - random scaling factor,
         β - data generation sampling
Output: s - synthetic rule-based classifier
1 $X, y \leftarrow makeClassificationData(n, m)$;      // make dataset using [38]
2 $X \leftarrow addRandomness(X, \alpha)$;             // add variability
3 $knn \leftarrow trainKNN(X, y, k = 3)$;              // define boundaries
4 $X' \leftarrow sampleData(X, \beta)$;                // generate data
5 $y' \leftarrow applyKNN(X', knn)$;                   // apply boundaries
6 $dt \leftarrow trainDecionTree(X', y')$;             // define global rules
7 function clf(x):
8     return $applyDecisionTree(x, dt)$;
9 return $s \leftarrow clf$;
```

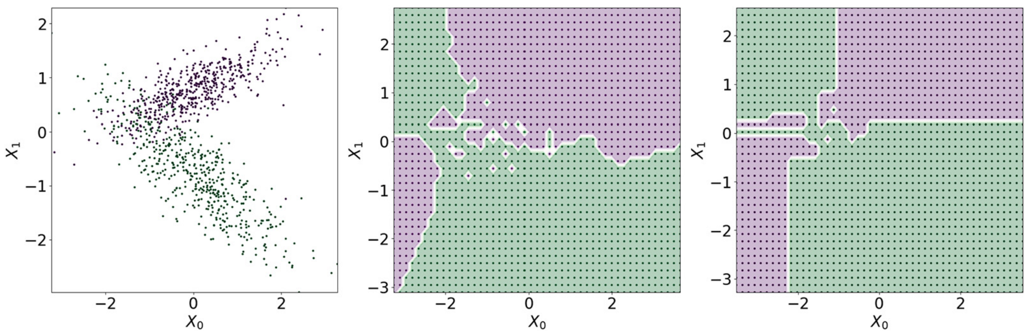

Fig. 2. *(Left)*. Original classification data X', y'. *(Center)*. Dataset X'', y' and decision boundary used for training the decision tree. *(Right)*. Same as (center) if lines 2-5 would have been removed from Algorithm 1. Best view in color. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

2. Covariance is added by multiplying by a random matrix, with uniformly distributed random numbers in $[-1, 1]$.
3. The clusters are then placed at random on the vertices of a hypercube in a m dimensional space.

Even though the *makeClassificationData* procedure would allow us to generate redundant or uninformative features, we only generate informative features and, as discussed in the following, we add uninformative features when performing experiments. We do not train the decision tree directly on the dataset $\langle X, y\rangle$ because this kind of dataset is generally quite simple, resulting in trivial decision trees consisting of few orthogonal rules separating the decision space. Therefore, since our objective is to stress *local* model-agnostic classifiers, we add further steps to increase the complexity of the classification dataset as follows. First, we add an α degree of randomness to X, augmenting the variability among the instances (line 2). Then, we train a *k-NN* classifier [27] with $k = 3$ to define the decision borders that the decision tree would have to discriminate (line 3). For each feature, we sample a dataset X' within the boundaries of X with a sampling rate of β (line 4).[8] We apply k-NN to classify X' and to obtain decisions y' (line 5). This dataset is used to train and generate the decision tree classifier dt (line 6).

Fig. 2 shows the benefits introduced by the SENECA-RB generator with an example in $m = 2$ features. The plot on the left depicts the classification data $\langle X, y\rangle$, the second plot reports the dataset X' and decision boundary y' provided as training for the decision tree, the third plot shows how the same information of the second plot if lines 2-5 would have been removed from Algorithm 1. It is clear how these lines introduce complexity to the decision boundary with the various turning points that allow the tree to pass from 79 decision rules to 214. This complexity allows to explore the effectiveness of the local explainer in finding the decision rule using the right features.

Given the decision tree dt, the synthetic classifier s classifies an instance x (line 8) by following the rule r on dt that *covers* x by checking the satisfied conditions from the root of s to a decision leaf which provides the outcome y. The rule r corresponds to the explanation e associated with the classification $y = s(x)$.

For instance, $\widetilde{e} = r = \{\{age \leq 25, income > 1500\} \rightarrow deny\}$.

[8] In the experiments we set $\alpha = 10$ and $\beta = 0.1$. These values are selected as consequence of a sensitivity analysis in which we vary $\alpha \in [1, 20], \beta \in [0.1, 0.9]$ with steps of 1 and 0.1, respectively. For each combination, we repeated 100 runs, and we observed the probability distributions of *knn* and *dt* internally generated and trained. The standard deviations of the average probabilities for both models is less than 0.001 for α and less than 0.005 for β. On the other hand, for α the null hypothesis of the Friedman test [40] is rejected for $p{-}value < 0.01$, while for β is not rejected for $p{-}value < 0.05$, in both cases for *knn* and *dt*. Hence, β does not impact the internal models of SENECA-RB, while α slightly affects their behavior, making the decision boundary of s more or less complex.

Algorithm 2: SENECA-RC(m, p).

```
Input  : $m$ - number of features, $p$ - binary operator prob
Output: $s$ - synthetic linear regression classifier
$f \leftarrow \langle\rangle$;                                         // || create random exception
for $i \in [0, \dots, m]$ do
    if $rnd() < p$ then
        $j \leftarrow selectFeature([0, \dots, m] - i)$;                // randomly select feature
        $op \leftarrow selectBinaryOperator()$;                         // randomly select operator
        $f \leftarrow f + op(x_i, x_j)$;                                // add operator to expression
    else
        $op \leftarrow selectUnaryOperator()$;                          // randomly select operator
        $f \leftarrow f + op(x_i)$;                                     // add operator to expression
function clf($x$):
    return $evaluate(x, f)$;
return $s \leftarrow clf$;
```

We perform experiments with SENECA-RB by generating random instances $X \in \mathcal{X}^{(m)}$ with m features in the same domain of the dataset X' on which s is trained. Experiments can be made more complex if u uninformative random features are added to X, i.e., $X \in \mathcal{X}^{(m+u)}$. In this case, the synthetic rule-based classifier s and the corresponding explanations would work only on the m real features, while the local model-agnostic explainer would consider all the $m + u$ features making harder the explanation task.

4.2. Linear classification model generator

The purpose of SENECA-RC is to generate a linear regression classifier s from which, given an instance x, is possible to extract the vector v of feature importance expressing the weight that each feature has for the decision $s(x)$.

As linear regression classifier s, we generate a function f using the method SENECA-RC described in Algorithm 2. The generation procedure is a simplification of the approach described in [41]. For each one of the m feature (loop 2-9) the method randomly decides[9] if a binary or unary operator is applied to the i^{th} feature. In the first case, the second feature j is selected (line 4). Then a binary operator is draw uniformly at random from $[x_1 + x_2, x_1 - x_2, x_1 * x_2, x_1/x_2, x_1^{x_2}]$ and added to the function with the two selected features (lines 5-6). In the second case a unary operator is draw uniformly at random from $[x, -x, sqrt(x), log(x), sign(x), sin(x), cos(x), tan(x), sinh(x), cosh(x), tanh(x), asin(x), acos(x), atan(x)]$ and added to the function with the selected feature (lines 8-9). The final function f in a simplified form constitutes the classifier s. Examples of generated functions are:

- $x_0^3 - 2x_1^2 + x_2$
- $sin(x_1) + x_1/x_0 + sign(x_2)$
- $-x_0 + x_2 + x_3 * x_5 - x_4 + x_4/x_5 + tanh(x4) + x_3/x_1$

Given an instance $x \in \mathbb{R}^{(m)}$, the outcome of s consists in evaluating[10] the function f in s with the values of x (line 11). Since we aim for a local explanation in form of a linear model, as assumed by local model-agnostic explainers returning features importance as explanations [25], the explanation e for x consist in the gradient[11] (represented as a weight vector v) of the closest point x^* on the classification boundary of s to x. More formally,

$$v = \langle f'_{x_i}(x_i^*) \rangle \quad i = 1 \dots m$$

where x^* is the closest point of x to the function $f = 0$. For instance, the feature importance explanation for the first function used as regression s in the list above is $v = \langle 3(x_0^*)^2, -4x_1^*, 1 \rangle$, i.e., if $x = \{2, 1, 5\}$, then the ground truth explanation for x is $\tilde{e} = v = \{x_0 = 12, x_1 = -8, x_2 = 1\}$.

Similarly to SENECA-RB, we experiment with SENECA-RC by generating random instances $X \in \mathcal{X}^{(m)}$ and also generating random instances $X \in \mathcal{X}^{(m+u)}$ with uninformative random features.

4.3. Image classifier generator

The purpose of the SENECA-IMG generator is to return an image classifier s from which, given an instance x (an image), is possible to extract the vector v of feature importance expressing the weight that each feature (pixel in this case) has for

[9] We set the binary operator probability p to 0.7 in our experiments.

[10] In order to ensure comparable results, we normalize between 0.0 and 1.0 the values returned by s, i.e., the co-domain of the function f. We accomplish this point by generating uniformly at random a sample of 1000 instances $x \in \mathbb{R}^{(m)}$, evaluating $f(x)$ and normalizing the resulting values. To do that, we force the classifier s to return 0.5 when x takes value outside from the domain of the function f.

[11] We force $f'_{x_i}(x_i^*)$ to return 0.0 if f'_{x_i} is not defined on x_i^*.

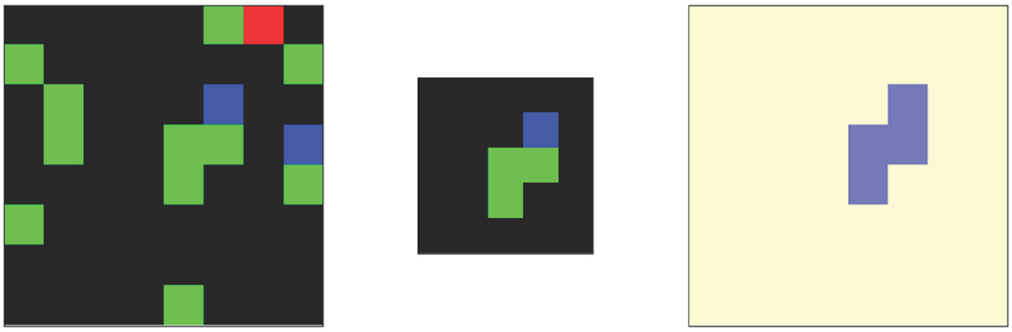

Fig. 3. *(Left)*. Example of image. *(Center)*. Pattern randomly generated. *(Right)*. Saliency map explaining the classifier outcome in recognizing the pattern.

Algorithm 3: SENECA-TXT(X, m, p).

```
Input  : $m$ - number of meaningful words, $X$ - training set,
         $p$ - probability of positive word
Output: $s$ - synthetic text classifier
1  $X \leftarrow preprocessing(X)$;                        // clean data
2  $W \leftarrow selectWords(X, m)$;                       // randomly select $m$ words
3  $\hat{W} \leftarrow \langle\rangle$;                    // initialize vector of weights
4  for $w_i \in W$ do                                      // for each word selected
5      $v \leftarrow rnd()$;                               // draw random number in [0,1)
6      if $rnd() < p$ then $\hat{W}_i \leftarrow +v$;      // add positive weight
7      else $\hat{W}_i \leftarrow -v$;                     // add negative weight
8  Function clf($x$):
9      $x' \leftarrow preprocessing(x)$;                   // clean data
10     $v \leftarrow 0$;                                   // initialize final value
11     for $x_i \in x$ do                                  // for each word in $x$
12         if $x_i \in \hat{W}$ then $v \leftarrow v + \hat{W}_i$;   // add word weight
13     return $v$;
14 return $clf$;
```

the classification of x by s. We restrict here to an image classifier for simple images of small size and without the presence of noise. However, it can be made arbitrarily more complex to increase the difficulty of the explanation problem. Let $h \times w$ pixels be the size of the images considered by the classifier. We assume that each image is segmented into a grid with *cells* of $h' \times w'$ pixels such that h' divides h and w' divides w. We consider images with a *black* background[12] and random cells colored in *cyan*, *green* or *blue*.[13] An example is reported in Fig. 3-*(left)*. We define a *pattern* as an image with $\hat{h} \times \hat{w}$ pixels that the classifier has to recognize ($\hat{h}$ divides h and $\hat{w}$ divides w). An example of pattern is reported in Fig. 3-*(center)*.

Given an image x and an image classifier s able to recognize a certain pattern a, the outcome of the classifier depends on the presence of the pattern a in the image x. The classifier s can recognize the presence/absence of a with a linear scan of the pixels in x with a window of the same dimension of a. In order for the pattern a to be recognized, both the shape and colors on the image must match with those of the pattern.[14] The local explanation e of s for recognizing a in x corresponds to the importance that each pixel has in revealing the presence of the pattern. Therefore, it can be modeled as a vector v of features (i.e., pixels) importance. The values v_i in v are equal to one if the pixel is part of the pattern, zero otherwise.[15] An example of feature importance vector visualized as a saliency map is reported in Fig. 3-*(right)*.

The experiments with SENECA-IMG can be carried on as follows. First, it is fixed the dimensions of the pattern. Then, it is generated s, and a dataset X of random images. Finally, the presence of the pattern recognized by s is guaranteed by randomly drawing it on half of the images of X.

4.4. Text classifier generator

The purpose of the SENECA-TXT generator is to return a text classifier s from which, given an instance x (text), is possible to extract the vector v of feature importance expressing the weight that each feature (word in this case) has for the classification of x by s. We restrict here to a simple text classifier using the positive/negative weights of words to classify

[12] All the three color channels turned to zero.
[13] On of the three color channels turned to one and the other two to zero.
[14] If present, black cells in the contour of the pattern contribute to the recognition and must be present also in the image for guaranteeing the match.
[15] If an image x does not contain the pattern then $v_i = 0$ for every pixel.

a text as positive or negative. However, like for the case of images, the classifier can be made arbitrarily more complex using n-grams (sets of words), or more semantically complex patterns. In the following, we provide the details of the text classifier generator described in Algorithm 3. Given a textual dataset X and the number of words m that the classifier has to use to perform the classification, SENECA-TXT works as follows. First, it cleans X (line 1, *preprocessing*) by removing stop-words, numbers and punctuation and separating every instance $x \in X$ into a list of words [42]. Then, it selects uniformly at random a set W of m words, among all the different words in X (line 2). For each word $w_i \in W$ (loop in lines 4-7), it randomly assigns a positive or negative weight v drawn uniformly at random in $[0, 1]$ to that word and stores it in the vector of weights $\hat{W}$. The synthetic text classifier s is defined (lines 8-13) using the weights $\hat{W}$.

Given a text x, the synthetic text classifier s, after cleaning x (line 9), assigns a positive or negative value to x depending on the words it contains (lines 11-12) with respect to the vector of weights $\hat{W}$. The local explanation e, in this case, corresponds to a vector v where, for each word contained in x, v_i is zero if the w_i word is not used by the classifier (i.e., is not in $\hat{W}$); otherwise it takes the value $\hat{W}_i$. An example is reported in the following. Given the weights used by the classifier as $\hat{W} = \langle (dog, 0.4), (frog, -0.3), (sun, 0.6), (green, -0.7), (jumps, 0.5) \rangle$ and the text $x =$ *"The quick brown fox jumps over the lazy dog"*, s returns the value 0.9 classifying the instance as positive. The ground truth explanation consists of $\widetilde{e} = v = \{dog = 0.4, frog = 0.0, sun = 0.0, green = 0.0, jumps = 0.5\}$.

The experiments with SENECA-TXT are performed as follows. Given a textual data X, X is split into training and test. The number of words to use m is fixed. Then, the classifier s is generated using the training set and used to classify the test set.

4.5. Discussion on synthetic explainable classifier generators

In this section, we discuss the strengths and weaknesses of SENECA generators. SENECA generators exploit *synthetic* random data and random weights to provide multiple and different transparent classifiers to deeply stress local explanation methods. An alternative approach to synthetic datasets would be to build transparent classifiers on *real* dataset. However, this kind of approach would tie the transparent classifier to a specific dataset without any possibility to play with the number of features or with the complexity of the explanation. On the other hand, these characteristics are allowed by the transparent classifiers generated by SENECA and are fundamental for stressing local explainers. In addition, testing the content of an explanation generated by a local explanation method only using real data would be limited by the availability of the real datasets and to the effectiveness of the transparent classifiers built on it. Moreover, a transparent classifier providing ground truth explanations based on real data can also be quite challenging to design concerning images (where we would need annotated images with relevant superpixels), and text (where we would need selected words together with their importance for the outcome). SENECA overcomes these limitations at the price of using randomness and synthetic data for building the transparent classifiers allowing access to ground truth explanations.

Another aspect to discuss is related to the fact that dimensionality can be a problem for classification models of AI systems. However, SENECA aims is to generate transparent classifiers having control of the data used to generate it. Therefore, the user can tune the dimensionality and stress the dimensionality aspect for the local explanation methods, as shown in the experimental section. As a consequence, SENECA does not suffer from large spaces or large models, but can help in testing the robustness of local explainers when explanations for classifiers dealing with large spaces are generated.

Finally, we point out that we could have formalized a unique notion of explanation and, as a consequence, a unique SENECA generator. However, we preferred to keep the different problems slightly separated even though with a similar formulation in order to be adherent with the state-of-the-art where each explanation method returns a slightly different type of explanation according to both the data type and problem solved.

5. Explanation evaluation

We measure the quality of an explanation e returned by a local model-agnostic explainer by observing its closeness with the true explanation $\widetilde{e}$ returned by a synthetic transparent classifier, i.e., the ground truth. Given e and $\widetilde{e}$, we adopts the following metrics [25]. If e and $\widetilde{e}$ are feature importance vectors, we measure the *quality* of e with respect of $\widetilde{e}$ using the *cosine similarity* of the two vectors:

$$q(e, \widetilde{e}) = cosine(e, \widetilde{e}) = \frac{\|e \cdot \widetilde{e}\|}{\|e\| \|\widetilde{e}\|}$$

where $e \cdot \widetilde{e}$ is the dot product, and $\|e\|$ calculates the L2-norm of e. The closer $q(e, \widetilde{e})$ is to 1, the higher the quality of e.

For decision rules, we model the explanation quality in two fashions. For the first metric, each rule is modeled as a vector expressing the presence/absence (one/zero) of a feature in the rule for each feature in the observed domain, and we measure the correct *features presence* in the explanation. For instance, if the domain consists of three features, given $r = \{x_0 > 0.3 \wedge x_1 \leq 0.7 \wedge x_1 > 0.2\} \rightarrow 1$, then the explanation is turned into $e = \{(x_0, 1), (x_1, 1), (x_2, 0)\}$. Hence, we measure the *quality* of rule explanations e regarding features correctness with respect to $\widetilde{e}$ using the *f1-score*:

$$q(e, \widetilde{e}) = f1{-}score(e, \widetilde{e}) = 2\frac{recall(e, \widetilde{e}) \cdot precision(e, \widetilde{e})}{recall(e, \widetilde{e}) + precision(e, \widetilde{e})}$$

where the *recall*$(e, \widetilde{e})$ indicates how many features retrieved by e are truly important in $\widetilde{e}$, and the *precision*$(e, \widetilde{e})$ indicates how many truly important features of $\widetilde{e}$ are correctly identified by e.

We highlight that the *f1-score* can also be used for measuring the quality of feature importance explanations by turning explanation vectors into binary vectors with zero if a feature has magnitude equals to zero, and one otherwise.

We name the second evaluation measure for rules *complete rule quality*. Every rule is turned into a vector where for each feature a_i in the observed domain are reported the values of the lower and upper bounds $v_i^{(l)}, v_i^{(u)}$. With respect to the previous example for $r = \{x_0 > 0.3 \wedge x_1 \leq 0.7 \wedge x_1 > 0.2\} \rightarrow 1$, for the complete rule quality we have $e = \{(x_0^{(u)}, \infty), (x_0^{(l)}, 0.3), (x_1^{(u)}, 0.7), (x_1^{(l)}, 0.2), (x_2^{(u)}, \infty), (x_2^{(l)}, -\infty), \}$. Thus, inspired by *k-mode* algorithm [43], we measure the *quality* of e with respect to $\widetilde{e}$ as:

$$q(e, \widetilde{e}) = \frac{1}{N_{\not\infty}} \sum_{i=1}^{|e|} \delta_{\varepsilon}(e_i, \widetilde{e}_i)$$

where

$$\delta_{\varepsilon}(e_i, \widetilde{e}_i) = \begin{cases} 1 & \text{if } |e_i - \widetilde{e}_i| \leq \varepsilon \wedge |e_i| \neq \infty \wedge |\widetilde{e}_i| \neq \infty, \\ 0 & \text{otherwise} \end{cases}$$

ε is the similarity threshold, and $N_{\not\infty}$ is the number of lower and upper bounds which are different from ∞ and $-\infty$ in both e and $\widetilde{e}$.[16] The more similar are two rules considering the thresholds of the upper and lower bounds, the higher is the quality. In the experiments, we report results using cosine similarity for explainers returning features importance on tabular data, images, and text; f1-score for explainers returning rules on tabular data and text; and complete rule quality[17] for rules on tabular data.

Concerning the definition of evaluation measures for explanation, it is worth to remark that in the literature there is not a shared definition of explanation. Consequently, there is no agreement on how to evaluate an explanation. Probably the best way to evaluate an explanation would be through a real experiment involving humans. However, since this is generally not possible, we propose SENECA. SENECA produces transparent synthetic classifiers that return ground truth explanations comparable with those of the local explanation methods in the state-of-the-art. Therefore, by construction, ground truth explanations returned by SENECA's classifiers have the same structural form of explanations returned by existing local explanation methods. As a consequence, the quality of an explanation can be evaluated using syntactical measures considering the presence/absence of features, weights of the features, features range, etc. In particular, we adopt the metrics defined above. Obviously, this kind of evaluation could be not generic enough, but it is parametric to future ways for evaluating the quality of an explanation with respect to its content. Indeed, the strength of the results presented in the following section lies in the context in which the experimentation is run and helps in reducing the very large spectrum of possible explanations and explanation methods. We highlight again that our proposal does not depend on a specific evaluation measure, but it focuses on the generation of synthetic transparent classifiers providing the ground truth for explanation evaluation.

6. Experiments

In this section, we employ SENECA generators for producing synthetic explainable classifiers. The objective is to evaluate, with massive experimentation involving various classifiers, the explanations returned by the following local model-agnostic explainers: LIME [28], MAPLE [30], SHAP [31], ANCHOR [36], BRL [37] (as implemented in [44]) and LORE [35]. We show how thanks to SENECA generators, it is possible to achieve an effective and statistically significant evaluation of these local explanation methods.[18] Besides evaluating the performance of local explanation methods, another application of SENECA could be to study to which extent interpretable classifiers [3] can recover local explanations. However, we remark that SENECA is meant to stress the behavior of *local* explanation methods, while a *global* interpretable classifier could not successfully address the task of retrieving a *local* explanation as it is not designed to solve it.

6.1. Experiments on tabular data

Using SENECA-RB and SENECA-RC, we study the behavior of the selected local model-agnostic explainers on tabular data. We differentiate the synthetic classifiers by changing the dimension of the instances X. In particular, we vary the number of (informative) features m, and the number of uninformative features u. The total number of features $m + u$ varies over $\{2, 4, 8, 16, 32, 64, 128\}$ and, for a fixed $m + u$ we set $m < \min\{32, m + u\}$. We generate 1000 rule-based classifiers and 1000 linear classifiers. For each synthetic classifier, we generate and explain 1000 instances.

[16] In this way, like for the Jaccard coefficient [27], the similarity is based only on the presence of a feature in a rule, and do not account for absence.

[17] We use $\varepsilon = 0.01$ in our experiments.

[18] The source code of SENECA and details of the initialization of the explainers in the various experiments are available at https://github.com/riccotti/SyntheticExplanationGenerator. LIME https://github.com/marcotcr/lime, MAPLE https://github.com/GDPlumb/MAPLE, SHAP https://github.com/slundberg/shap, ANCHOR https://github.com/marcotcr/anchor, BRL https://github.com/rulematrix/rule-matrix-py, LORE https://github.com/riccotti/LORE.

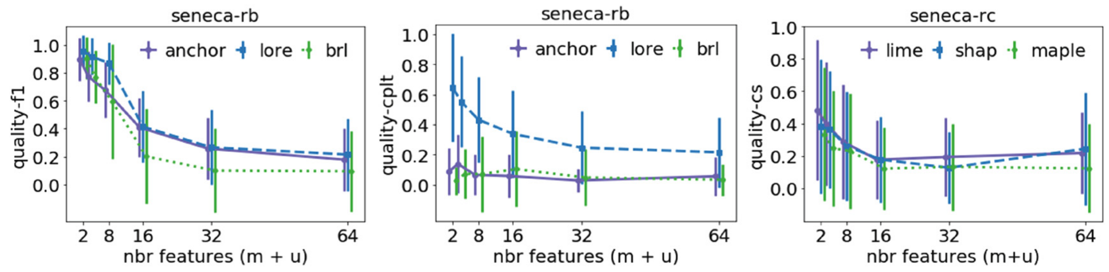

Fig. 4. Quality of rule-based explanations evaluated with f1-score *(left)* and complete rule quality *(center)*, and feature importance explanations evaluated with cosine similarity *(right)* on tabular data varying the number of features m used by the classifiers generated with SENECA-RB and SENECA-RC, respectively.

Table 1
Quality of explanations for values in Fig. 4 up to $m+u=512$ expressed as f1-score *(left)*, complete rule *(center)*, and cosine similarity *(right)*.

	f1			*cplt*			*cs*		
m	ANCHOR	LORE	BRL	ANCHOR	LORE	BRL	LIME	SHAP	MAPLE
2	.90 ± .1	**.96 ± .1**	.90 ± .1	.09 ± .2	**.65 ± .36**	.02 ± .1	**.48 ± .4**	.38 ± .4	.33 ± .4
4	.77 ± .1	**.92 ± .1**	.77 ± .1	.14 ± .2	**.55 ± .30**	.07 ± .2	**.40 ± .3**	.36 ± .3	.25 ± .3
8	.68 ± .2	**.87 ± .1**	.59 ± .4	.07 ± .1	**.43 ± .3**	.07 ± .3	**.29 ± .3**	.26 ± .3	.23 ± .3
16	.41 ± .2	**.42 ± .2**	.20 ± .3	.06 ± .1	**.34 ± .3**	.10 ± .3	.25 ± .2	**.18 ± .2**	.12 ± .2
32	.26 ± .2	**.27 ± .2**	.10 ± .3	.03 ± .1	**.25 ± .2**	.05 ± .2	**.24 ± .2**	.13 ± .2	.13 ± .2
64	.18 ± .2	**.21 ± .2**	.09 ± .2	.06 ± .1	**.22 ± .2**	.03 ± .1	.22 ± .2	**.24 ± .3**	.12 ± .2
128	.16 ± .3	**.19 ± .3**	.07 ± .3	.25 ± .4	**.34 ± .4**	.30 ± .4	.19 ± .2	.19 ± .32	.11 ± .2
256	.15 ± .2	**.17 ± .2**	.05 ± .2	.05 ± .1	**.40 ± .3**	.11 ± .2	.16 ± .2	**.26 ± .3**	.11 ± .3
512	.13 ± .2	**.16 ± .2**	.04 ± .3	.01 ± .1	**.22 ± .3**	.03 ± .1	.13 ± .3	**.27 ± .3**	.10 ± .4

Fig. 4 reports the quality of rule-based explanations evaluated with f1-score *(left)* and complete rule quality *(center)*, and feature importance explanations evaluated with cosine similarity *(right)* on tabular data varying the number of features $m+u \leq 64$ (also considering a part of uninformative features u) used by classifiers generated with SENECA-RB and SENECA-RC, respectively. Every point is the average quality among various instances and classifiers (the higher, the better). The vertical bar represents the standard deviation of the quality (the smaller, the better). We add a small misalignment of the vertical bars for the sake of readability. Table 1 reports the values of the lines in Fig. 4 plus explanation quality up to $m+u=512$.

With respect to the f1-score quality ANCHOR and LORE achieve comparable performance (both in terms of mean and standard deviation) and with a similar decrease when the number of features $m+u$ increases for the presence of the features in the rules. LORE slightly performs better than ANCHOR and BRL when $m+u \leq 4$. Then, for $m+u \geq 16$, the performance of BRL are significantly lower than those of the others. A weakness of BRL is that it is not able to return an explanation within sixty seconds when $m+u > 6$, therefore we do not report its performance. LORE performs better than ANCHOR when $m+u \leq 8$. After this point they maintains comparable f1-score quality with LORE having slightly higher quality at the price of a slightly lower stability. On the other hand, with respect to complete rule quality, Fig. 4 *(center)*, LORE markedly outperforms both ANCHOR and BRL independently from the number of features. This implies that despite ANCHOR is comparable to LORE in terms of the features detected for composing a rule-based explanation, the upper and lower bound identified by LORE are much more accurate than those of ANCHOR. The markedly better performance of LORE with respect to ANCHOR and BRL in terms of complete rule quality are probably due to the interpretable surrogate model internally adopted for finding the explanation rule. ANCHOR builds a unique rule trying to identify fundamental conditions focusing on coverage and precision. However, perhaps this induces a too restricted set of conditions with a unique perspective for the rule searched. BRL builds a Bayesian rules list and selects the most promising rule according to the probabilities detected. Inaccurate probabilities due to the noise and the number of features can negatively impact on the construction of the rule lists and on the final selection. Finally, LORE adopts a decision tree which aims at covering the overall local decision boundary and also find counter-factual explanations to understand the factual reasons. This double objective in search of the explanation can help LORE in finding the most correct explanation in terms of conditions in the rule.

On the other hand, with respect to explainers based on feature importance, the three approaches have comparable performance with LIME performing better than the others when $m+u < 16$, and SHAP when $m+u \geq 64$. SHAP is also the explainer with the lowest deviation among the explanation qualities. In addition, when the number of features $m+u$ grows, SHAP, with alternating results, shows the best performance. MAPLE explanation qualities are always shallow, even though when the number of features is low and therefore should be easier to retrieve the correct explanation. We highlight how the explanation quality measured with cosine similarity is lower than the one measured with f1-score. This happens because the evaluation through cosine similarity makes the explanation task more difficult since not only the presence/absence of a feature is considered, but also its local importance. Finally, we highlight that all the explainers have not good performance (with explanation quality around 0.2) when $m+u \leq 16$.

In Fig. 5, given the total number of features $m+u$, we show the effect of varying the number of informative features m on the quality of rule-based explainers evaluated with f1-score *(top)* and complete rule *(center)*, and on the quality of

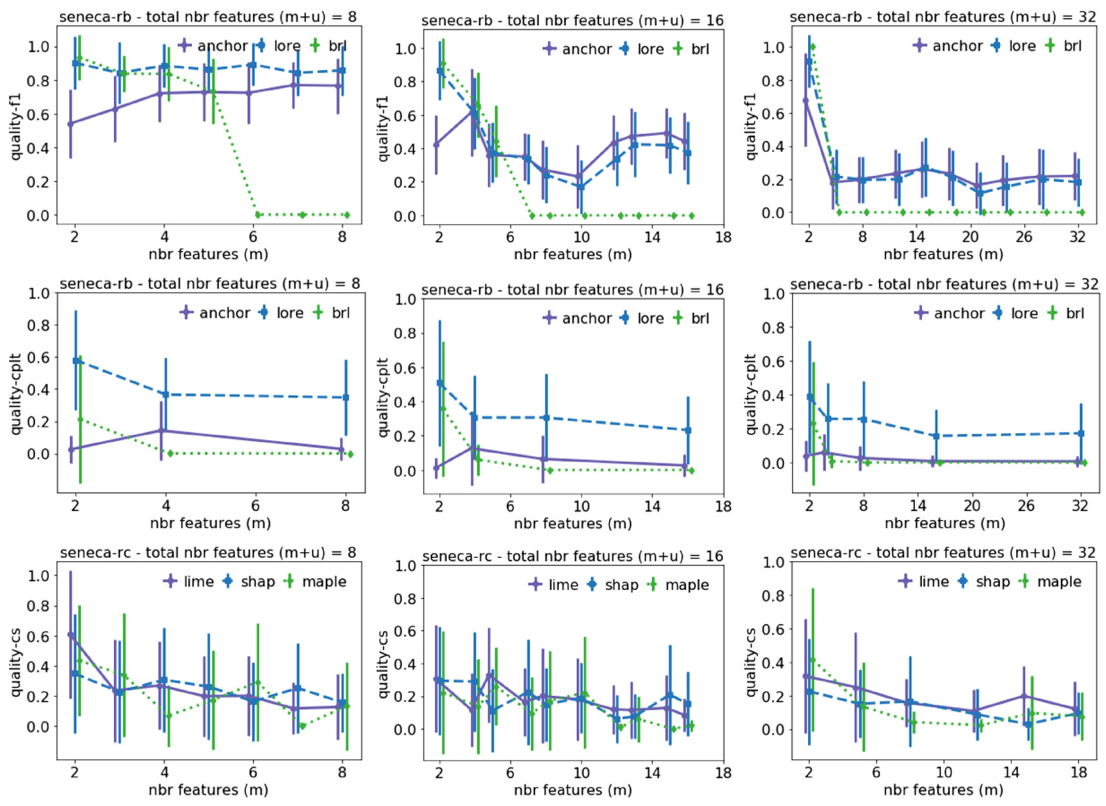

Fig. 5. Quality of rule-based explanations evaluated with f1 *(top)* and complete rule *(center)*, and feature importance explanation evaluated with cosine similarity *(bottom)* on tabular data varying the number of features m given the total number of features $m+u$ used by the transparent classifiers generated with SENECA-RB and SENECA-RC.

explainers based on feature importance evaluated with cosine similarity *(bottom)* on tabular data. The higher m, the higher is the correspondence between all the features and the informative features. With respect to rule-based explainers (Fig. 5 *(top)* and *(center)*), when $m+u=8$, i.e., 1st plot, there is not a consistent variation on the performance when varying m: BRL and ANCHOR are the worst explainers for *f1* and *cplt*, respectively. When $m+u=16$, i.e., 2nd plot, we observe a drop in the performance when $6 \leq m \leq 12$ for the *f1*. It means that, when there are few or many informative features out of 16, the explainers are able to identify them, while for cases in between they fail. The same pattern appears for $m+u \geq 32$, i.e., 3^{rd} plot,[19] when $m=2$. It means that the explainers fail to understand the important features among the available ones, whether they are informative or uninformative. For *cplt* we do not observe the same variations as for *f1*, because LORE explanations' quality remains stably higher varying m for both $m+u=16$ and $m+u=32$. On the other hand, with respect to explainers returning features importance (Fig. 5-*(bottom)*), we observe in every plot that a decrease in the number of uninformative features causes a decrease of the quality. It means that the linear explainers can capture the correct coefficient of features only when there are a few important features. This appears to be the main limitation for all the explainers with respect to tabular data. Finally, we highlight that in every plot, there are no changes in the deviations when varying the number of features.

6.2. Experiments on images

We analyze the behavior of LIME, SHAP and MAPLE in explaining the classifications of synthetic image classifiers generated using SENECA-IMG. We set the image dimension in terms of number of pixels as $h=w=32$ and the cell dimension as $h'=w'=4$. We generate different synthetic image classifiers by varying the dimension $\hat{h} \times \hat{w}$ of the pattern a that the classifier is able to recognize with $\hat{h}=\hat{w}=m$ and $m \in \{8, 12, 16, 20, 24, 28, 32\}$. For each dimension, we generate 20 different patterns, and for each pattern, we generate a dataset X of 1000 random images, of which half of them contain the pattern to recognize. We evaluate the explanation quality using the f1-score [45]: the higher is the overlap between pixels of the ground truth (corresponding to the pattern), and pixels of the explanation, the better is the explanation. Fig. 6 shows explanations in the form of saliency maps with respect to the image and to the classifier that recognizes the pattern

[19] $m+u=64$ not reported in the paper but available in the repository.

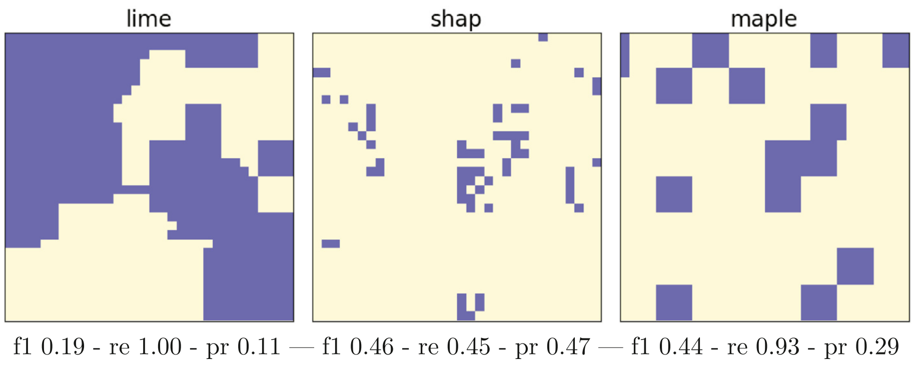

Fig. 6. Pixel importance explanations in forms of saliency maps for a synthetic transparent classifier able to recognize the pattern in Fig. 3-*(center)* returned by LIME, SHAP and MAPLE for the image in Fig. 3-*(left)*. The values are the explanation quality in terms of f1-score, recall and precision.

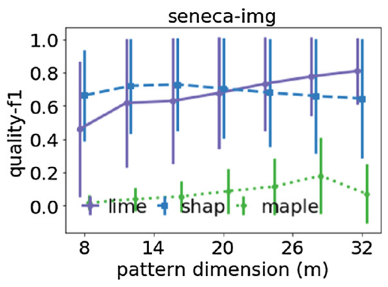

Fig. 7. Explanation quality of pixel importance explanations evaluated with f1-score varying the pattern dimension $m = \hat{h} = \hat{w}$ used by the synthetic image classifiers generated with SENECA-IMG *(left)*, and corresponding values *(right)*.

Table 2
Explanation quality values for lines in Fig. 7 in terms of f1-score.

m	LIME	SHAP	MAPLE
8	$.46 \pm .41$	$\mathbf{.66 \pm .27}$	$.02 \pm .04$
12	$.62 \pm .39$	$\mathbf{.72 \pm .29}$	$.04 \pm .07$
16	$.66 \pm .35$	$\mathbf{.71 \pm .30}$	$.08 \pm .11$
20	$.68 \pm .34$	$\mathbf{.70 \pm .30}$	$.09 \pm .14$
24	$\mathbf{.73 \pm .28}$	$.68 \pm .33$	$.11 \pm .17$
28	$\mathbf{.78 \pm .24}$	$.66 \pm .34$	$.18 \pm .23$
32	$\mathbf{.81 \pm .20}$	$.64 \pm .36$	$.07 \pm .18$

reported in Fig. 3. Under each figure is reported the f1-score, recall, and precision. We notice how LIME's explanation covers all the areas of the pattern (recall 1.0) but also other areas that are not used by the classifier (that implies the low precision). MAPLE seems to better delineate the pattern with respect to LIME, but it also adds to the explanation of some wrong cells. SHAP is more conservative in signaling important pixels, and the high precision leads to the highest f1-score in the example.

In Fig. 7 we show the explanation quality of pixel importance explanations varying the pattern dimension $m = \hat{h} = \hat{w}$ used by the synthetic image classifiers generated with SENECA-IMG. The table on the right reports the corresponding values. SHAP is the best image explainer with respect to both average explanation quality and deviation up to $m = 20$. On the other hand, LIME performance improves with the increasing dimension of the pattern, and for $m \geq 24$ LIME becomes better than SHAP in providing the explanation as saliency maps. Moreover, how highlighted in Table 2, LIME deviation markedly decreases. This behavior is probably due to *(i)* the superpixel technique adopted by LIME to segment the image, and *(ii)* LIME's attitude to overestimate important areas leading to a high recall (like in the example). As a consequence, when the relevant area grows and becomes of the same dimension of the image, LIME performance turns to be the best one, Finally, MAPLE is constantly the worst explainer.

6.3. Experiments on texts

In the following, we analyze the explanation returned by feature importance based explainers for synthetic text classifiers generated using SENECA-TXT. As textual dataset X we use *20 Newsgroups*.[20] We split it into train and test in line with the

[20] http://qwone.com/~jason/20Newsgroups/.

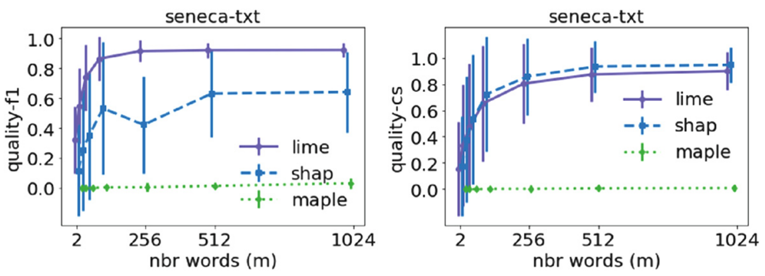

Fig. 8. Explanation quality of word importance explanations evaluated with f1-score *(left)* and cosine similarity *(right)* varying the number of words *m* used by the synthetic text classifiers generated with SENECA-TXT.

Table 3
Explanation quality values for lines in Fig. 8 in terms of f1-score *(left)* and cosine similarity *(right)*.

m	LIME *f1*	SHAP *f1*	MAPLE *f1*	LIME *cs*	SHAP *cs*	MAPLE *cs*
10	**.32** ± **.22**	.12 ± .31	.00 ± .00	.15 ± .36	**.18** ± .38	.00 ± .00
25	**.55** ± **.25**	.26 ± .40	.00 ± .00	.33 ± .47	**.38** ± .48	.00 ± .00
50	**.74** ± **.22**	.35 ± .43	.00 ± .00	.47 ± .48	**.53** ± .49	.00 ± .00
100	**.86** ± **.15**	.53 ± .44	.00 ± .02	.65 ± .44	**.72** ± .44	.00 ± .02
250	**.91** ± **.07**	.42 ± .32	.00 ± .03	.80 ± .31	**.86** ± .30	.00 ± .03
500	**.92** ± **.05**	.63 ± .29	.01 ± .02	.87 ± .21	**.93** ± .20	.01 ± .02
1000	**.92** ± **.05**	.64 ± .27	.03 ± .04	.90 ± .15	**.95** ± .14	.01 ± .02

literature and, after the preprocessing, we consider 1056 words. We generate different synthetic text classifiers by varying the number of words m that the classifier uses to label a text as positive or negative in $\{10, 25, 50, 100, 250, 500, 1000\}$. The selection of m words to build a text classifier is repeated ten times and, for each text classifier, we classify 1000 texts of the test set. We evaluate the text explanation quality using both the f1-score and the cosine similarity.

Fig. 8 reports the explanation quality of word importance explanations evaluated with *f1-score* (left) and *cosine similarity* (right) varying the number of words m used by the synthetic text classifiers generated with SENECA-TXT. We report in Table 3 the values of the lines in Fig. 8. The two plots highlight a different behavior. LIME returns the best explanations with respect to which are the words identified. On the other hand, SHAP appears to be the best in understanding, which is the level of importance of these words. Thus, the coefficients of a linear regressor are better to identify *which* are the most important features, while the Shapely values are better for identifying the *level of importance* of these features. Perhaps a combined approach would lead to an overall more correct explanation. In both cases, the explanations reach higher quality when synthetic classifiers use a larger set of important words. Therefore, the higher is the number of important words used by a text classifier with respect to the vocabulary considered, the better is the explanation quality. Another aspect that is worth mentioning is that with respect to both the evaluation measures, LIME extracts more stable explanations than SHAP (lower standard deviations). Finally, like for images, also for text MAPLE is not a good explainer. We highlight that, similarly to the results observed for images, LIME's *f1-scores* are largely supported by a very high recall that is on average of 0.99 versus the 0.38 of SHAP. However, differently from what observed for images, LIME performance is guaranteed not only by the high recall but also by an average precision of 0.66, while SHAP's average precision is 0.62.

6.4. Statistical comparison of explanation methods

In the following, we show the statistical significance of the results previously discussed. The non-parametric Friedman test [40] compares the average ranks of explanation methods over multiple classifiers with respect to an evaluation measure, i.e., the average explanation quality measured as f1-score, complete rule quality, or cosine similarity. The null hypothesis of the Friedman test that all methods are equivalent is rejected (p-*value*<0.001) for all the experiments using SENECA-RB, SENECA-RC, SENECA-IMG and SENECA-TXT considering all the synthetic classifiers and datasets. Moreover, we represent the comparison of the ranks of all explainers against each other in Fig. 9 with the critical difference plots (see [46] for details). Critical difference diagrams show the results of a statistical comparison of the performance of all the local explanation methods. In these diagrams, the explanation methods, represented by vertical+horizontal lines, are displayed from left to right in terms of the average rank obtained for the various evaluation measures and experiments (1 indicates the best performer, 3 indicates the worst performer). Horizontal bold lines connect the methods producing statistically equivalent performance according to a post-hoc Nemenyi test. We observe that for rule-based explainers LORE is significantly better than the others for both correct feature presence (f1-score), and complete rule quality. With respect to feature importance explanation for tabular data and images we can choose independently from LIME and SHAP. SHAP is the best choice for small and scattered relevant areas, while LIME should be preferred when it is known that the relevant areas are consistently large. With respect to text, LIME is statistically the best choice when looking to the words used, while if we look at the weight of the words SHAP is the best choice.

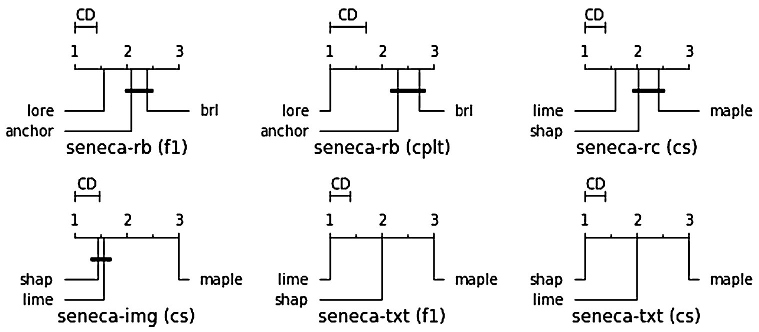

Fig. 9. Critical difference diagrams using the Post-hoc Nemenyi test for rule-based explanation (1st and 2nd plots) and feature importance (3rd plot) for tabular data, for pixel importance (4th plot) for images, and word importance (5th and 6th plots) for text, respectively. Every critical difference diagram shows the results of a statistical comparison of the performance of all the local explanation methods for a specific evaluation measure and problem setting. The explanation methods are represented by vertical+horizontal lines and are displayed from left to right in terms of the average rank obtained. The post-hoc Nemenyi test ties with a horizontal bold line pair of methods for which the ranking of performance is not statistically significant.

It is worth to underline the similarity between the final aim of the presented work with model selection consistency [47] as SENECA could be exploited for explanation model selection. Indeed, thanks to the proposed approach it is possible to evaluate the performance of various local explanation methods with respect to multiple and different transparent classifiers that can be "similar" (e.g., in terms of the number of features) to the real and not interpretable one a user could be interested in explaining.

7. Conclusion

In this paper, we have proposed SENECA a family of synthetic explainable classifier generators for tabular data, images, and text for which the explanations of individual decisions are available by design such that it is possible to evaluate the performance of local model-agnostic explanation methods with ground truth. A deep experimentation on state-of-the-art local explainers has shown how it is possible to estimate the quality of the explanations returned and enhance the strengths and weaknesses of the explanation methods. With the support of statistical tools, the proposed methodology provides reliable indexes that capture the quality of local explanations and describe if an explanation method focuses on the ground truth. Experiments with SENECA generators highlighted the following weaknesses. With respect to tabular data, results show that local explainers suffer in finding good explanations when the explanation is formed by many relevant features: they fail in considering all the relevant features and in assigning the correct weight to them. Concerning images, they lack in returning accurate saliency maps as the explanations provided are either too large (LIME) or too small and scattered (SHAP). Finally, regarding text, the explainers are not good when the classifiers focus on a restricted set of words with respect to the vocabulary.

As future work, we plan to extend the benchmarking to the growing literature of local model-agnostic explainers. Moreover, we would like to overcome some deficiencies of the current version of SENECA-IMG and SENECA-TXT by improving the level of complexity of the synthetic transparent image and text classifiers generated. Concerning images, we can produce transparent classifiers for more sophisticated images (e.g., landscapes, faces, medical records, etc.) able to recognize articulated multiple patterns (e.g., the sun position with respect to the landscape, the distance between eyes). Similarly, we can generate text classifiers accounting for more complex data structures like n-grams, sentences, etc. Finally, we would like to devise a synthetic transparent classifier generator with a human-in-the-loop [48], such that the ground truth explanations can somehow be extrapolated from the interaction with a real person instead of from pure random processes.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work is partially supported by the European Community H2020 programme under the funding schemes: INFRAIA-01-2018-2019 Res. Infr. G.A. 871042 *SoBigData++* (sobigdata.eu), G.A. 952026 *HumanE AI Net* (humane-ai.eu), G.A. 825619 *AI4EU* (ai4eu.eu), G.A. 952215 *TAILOR* (tailor.eu).

References

[1] A.C.M. Council, US public policy, statement on algorithmic transparency and accountability, Commun. ACM (2018).

[2] R. Goebel, A. Chander, K. Holzinger, F. Lecue, Z. Akata, S. Stumpf, P. Kieseberg, A. Holzinger, Explainable AI: the new 42?, in: International Cross-Domain Conference for Machine Learning and Knowledge Extraction, Springer, 2018, pp. 295–303.
[3] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, D. Pedreschi, A survey of methods for explaining black box models, ACM Comput. Surv. (CSUR) 51 (5) (2018) 93.
[4] D. Pedreschi, F. Giannotti, R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, Meaningful explanations of black box AI decision systems, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 9780–9784.
[5] S. Wachter, B. Mittelstadt, L. Floridi, Why a right to explanation of automated decision-making does not exist in the general data protection regulation, Int. Data Privacy Law 7 (2) (2017) 76–99.
[6] G. Malgieri, G. Comandé, Why a right to legibility of automated decision-making exists in the general data protection regulation, Int. Data Privacy Law 7 (4) (2017) 243–265.
[7] T. Miller, Explanation in artificial intelligence: insights from the social sciences, Artif. Intell. 267 (2019) 1–38.
[8] F. Doshi-Velez, B. Kim, Towards a rigorous science of interpretable machine learning, arXiv:1702.08608, 2017.
[9] C. Molnar, Interpretable Machine Learning, LeanPub, Victoria, Canada, 2018.
[10] H.J. Escalante, S. Escalera, I. Guyon, et al., Explainable and Interpretable Models in Computer Vision and Machine Learning, Springer, Berlin, Heidelberg, 2018.
[11] A.A. Freitas, Comprehensible classification models: a position paper, ACM SIGKDD Explor. Newsl. 15 (1) (2014) 1–10.
[12] J.V. Frasch, A. Lodwich, F. Shafait, T.M. Breuel, A Bayes-true data generator for evaluation of supervised and unsupervised learning methods, Pattern Recognit. Lett. 32 (11) (2011) 1523–1531.
[13] Y. Pei, O. Zaïane, A synthetic data generator for clustering and outlier analysis, Tech. Rep. 1 (1) (2006) 1.
[14] R. Guidotti, R. Trasarti, M. Nanni, Tosca: two-steps clustering algorithm for personal locations detection, in: Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, New York, USA, 2015, p. 38.
[15] E. Cesario, G. Manco, R. Ortale, Top-down parameter-free clustering of high-dimensional categorical data, IEEE Trans. Knowl. Data Eng. 19 (12) (2007) 1607–1624.
[16] R. Guidotti, A. Monreale, M. Nanni, F. Giannotti, D. Pedreschi, Clustering individual transactional data for masses of users, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, USA, 2017, pp. 195–204.
[17] M. Bouguessa, A practical approach for clustering transaction data, in: International Workshop on Machine Learning and Data Mining in Pattern Recognition, Springer, Berlin, Heidelberg, 2011, pp. 265–279.
[18] A. Lancichinetti, S. Fortunato, Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities, Phys. Rev. E 80 (1) (2009) 016118.
[19] D. Hric, R.K. Darst, S. Fortunato, Community detection in networks: structural communities versus ground truth, Phys. Rev. E 90 (6) (2014) 062805.
[20] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, Knowl. Inf. Syst. 42 (1) (2015) 181–213.
[21] G. Rossetti, L. Pappalardo, S. Rinzivillo, A novel approach to evaluate community detection algorithms on ground truth, in: Complex Networks VII, Springer, Berlin, Heidelberg, 2016, pp. 133–144.
[22] T. Van den Bulcke, K. Van Leemput, B. Naudts, P. van Remortel, H. Ma, A. Verschoren, B. De Moor, K. Marchal, Syntren: a generator of synthetic gene expression data for design and analysis of structure learning algorithms, BMC Bioinform. 7 (1) (2006) 43.
[23] T.S. Buda, T. Cerqueus, J. Murphy, M. Kristiansen Rex, Extrapolating relational data in a representative way, in: British International Conference on Databases, Springer, Berlin, Heidelberg, 2015, pp. 95–107.
[24] Y. Jia, J. Bailey, K. Ramamohanarao, C. Leckie, X. Ma, Exploiting patterns to explain individual predictions, Knowl. Inf. Syst. 1 (2019) 1–24.
[25] Y. Jia, J. Bailey, K. Ramamohanarao, C. Leckie, M.E. Houle, Improving the quality of explanations with local embedding perturbations, in: Proceedings of the 25nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, vol. 1, ACM, New York, USA, 2019, p. 1.
[26] Y. Zhang, K. Song, Y. Sun, S. Tan, M. Udell, "Why should you trust my explanation?" Understanding uncertainty in lime explanations, arXiv preprint, arXiv:1904.12991, 2019.
[27] P.-N. Tan, Introduction to Data Mining, Pearson Education India, India, 2018.
[28] M.T. Ribeiro, S. Singh, C. Guestrin, Why should I trust you?: explaining the predictions of any classifier, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, USA, 2016, pp. 1135–1144.
[29] K. Simonyan, A. Vedaldi, A. Zisserman, Deep inside convolutional networks: visualising image classification models and saliency maps, arXiv:1312.6034, 2013.
[30] G. Plumb, D. Molitor, A.S. Talwalkar, Model agnostic supervised local explanations, in: Advances in Neural Information Processing Systems, Curran Associates, Inc., New York, USA, 2018, pp. 2515–2524.
[31] S.M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: Advances in Neural Information Processing Systems, Curran Associates, Inc., New York, USA, 2017, pp. 4765–4774.
[32] A. Shrikumar, P. Greenside, A. Shcherbina, A. Kundaje, Not just a black box: learning important features through propagating activation differences, arXiv:1605.01713, 2016.
[33] M. Sundararajan, A. Taly, Q. Yan, Axiomatic attribution for deep networks, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR.org., Sydney, Australia, 2017, pp. 3319–3328.
[34] R. Guidotti, A. Monreale, S. Matwin, D. Pedreschi, Black box explanation by learning image exemplars in the latent feature space, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2019, pp. 189–205.
[35] R. Guidotti, A. Monreale, F. Giannotti, D. Pedreschi, S. Ruggieri, F. Turini, Factual and counterfactual explanations for black box decision making, IEEE Intell. Syst. 34 (6) (2019) 14–23.
[36] M.T. Ribeiro, S. Singh, C. Guestrin Anchors, High-precision model-agnostic explanations, in: AAAI, AAAI Press, New York, 2018, pp. 1527–1535.
[37] H. Yang, C. Rudin, M. Seltzer, Scalable Bayesian rule lists, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR.org., Sydney, Australia, 2017, pp. 3921–3930.
[38] I. Guyon, Design of experiments of the nips 2003 variable selection benchmark, in: NIPS 2003 Workshop on Feature Extraction and Feature Selection, NIPS, New York, USA, 2003, p. 1.
[39] L. Breiman, Classification and Regression Trees, Routledge, New York, 2017.
[40] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, J. Am. Stat. Assoc. 32 (200) (1937) 675–701.
[41] A. Klimke, Randexpr: A Random Symbolic Expression Generator, 2003.
[42] C. Manning, P. Raghavan, H. Schütze, Introduction to information retrieval, Nat. Lang. Eng. 16 (1) (2010) 100–103.
[43] N. Sharma, N. Gaud, K-modes clustering algorithm for categorical data, Int. J. Comput. Appl. 127 (1) (2015) 46.
[44] Y. Ming, H. Qu, E. Bertini, Rulematrix: visualizing and understanding classifiers with rules, IEEE Trans. Vis. Comput. Graph. 25 (1) (2018) 342–352.
[45] R. Guidotti, A. Monreale, L. Cariaggi, Investigating neighborhood generation for explanations of image classifiers, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, Berlin, Heidelberg, 2019, pp. 136–149.
[46] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (Jan 2006) 1–30.
[47] P. Zhao, B. Yu, On model selection consistency of lasso, J. Mach. Learn. Res. 7 (Nov 2006) 2541–2563.
[48] B. Krarup, M. Cashmore, D. Magazzeni, T. Miller, Model-Based Contrastive Explanations for Explainable Planning, 2019.