A Systematic Review on Model-agnostic XAI Libraries*

Jesus M. Darias, Belén Díaz-Agudo[0000−0003−2818−027X], and Juan A. Recio-Garcia[0000−0001−8731−6195]

Department of Software Engineering and Artificial Intelligence
Instituto de Tecnologías del Conocimiento
Universidad Complutense de Madrid, Spain
{jdarias,belend,jareciog}@ucm.es

Abstract. During the last few years the topic explainable artificial intelligence (XAI) has become a hotspot in the ML research community. Model-agnostic interpretation methods propose separating the explanations from the ML model, making these explanation methods reusable through XAI libraries. In this paper we have reviewed some selected XAI libraries and provide examples of different model agnostic explanations for the same black box model with the same training data. The context of the research conducted in this paper is the iSee project [1] that will show how users of Artificial Intelligence (AI) can capture, share and re-use their experiences of AI explanations with other users who have similar explanation needs.

Keywords: XAI, libraries, model agnostic models

1 Introduction

Interpretability and trust have became a requirement for black box models applied to real world tasks like diagnosis or decision making processes. At a high level, the literature distinguishes between two main approaches to interpretability: *model-specific* (also called transparent or white box) models and *model-agnostic* (post-hoc) surrogate models to explain black box models [9, 10, 15]. Transparent models are ones that are inherently interpretable by users. Consequently, the easiest way to achieve interpretability is to use algorithms that create interpretable models, such as decision trees, simple nearest-neighbour models or linear regression. However, the best performing models are often not interpretable, or they are partially interpretable [7]. However, it is a permanent challenge to ensure a high accuracy of a model while maintaining a sufficient level

* Supported by the Horizon 2020 Future and Emerging Technologies (FET) programme of the European Union through the iSee project (CHIST-ERA-19-XAI-008 - PCI2020-120720-2) and the Spanish Committee of Economy and Competitiveness (TIN2017-87330-R).

[1] Intelligent Sharing of Explanation Experience by Users for Users https://isee4xai.com/

of comprehensibility. Model-agnostic interpretation methods propose separating the explanations from the ML model. Although the main advantage is flexibility, as the interpretation methods can be applied to any model, some authors consider this type of post-hoc explanations as limited *justifications* because they are not linked to the real reasoning process occurring in the black box. The context of the research conducted in this paper is the *iSee project* that aims to provide a unifying platform where personalized explanations are created by reasoning with Explanation Experiences using Case-based reasoning (CBR). This is a very challenging, long term goal as we want to capture complete user centered explanation experiences on complex explanation strategies. Our proposal relies on an ontology to help to the knowledge intensive representation of previous experiences, different types of users and explanation needs, characterization of the data, the black box model and the contextual properties of the application domain and task. We aim to be able to recommend what explanation strategy better suits an explanation situation. One of the first tasks in the iSee project is to be able to characterize the existing XAI libraries. Explainers of these libraries will be the building blocks of our library of reusable explanation strategies that will be described using the unified terminology defined by the ontology.

In this position paper, we have reviewed some existing libraries of eXplainable Artificial Intelligence with the goal of understanding the capabilities of the different choices. These libraries are: Interpret, Alibi, Aix360, Dalex and Dace. We have compared different options to explain the same black box model with the same training data and the most relevant explanation methods, namely: Local Interpretable Model-Agnostic Explanations (LIME), Anchors, Shapley Additive Explanations (SHAP), Partial Dependence Plots (PDPs), Accumulated Local Effects (ALE) and counterfactual explanations.

Section 2 describes these explanation methods that focus our analysis. In section 3, we present the methodology designed to compare the libraries and the implemented explainers using a common use case. This section defines the variables used to perform a quantitative analysis of the libraries that is presented in Section 4. However, the XAI methods are analysed through a qualitative evaluation described in Section 5. Finally, Section 6 concludes the paper by discussing and comparing the libraries.

2 XAI methods

This section presents the most relevant explainers provided by the libraries analyzed in this work, whose availability is summarized in Table 1.

Local Interpretable Model-Agnostic Explanations (LIME) [11] is one of the most popular basic explainers. LIME attempts to understand the model by perturbing the input of data samples and understanding how the predictions change. The intuition to local interpretability is to determine which feature changes will have the most impact on the prediction.According to its authors, the algorithm fulfils the desirable aspects of a model-agnostic explanation system regarding flexibility. The LIME interpretation method can

	LIME	Anchors	SHAP	PDP	ALE	Counterfactuals	CEM
Interpret	✓		✓	✓			
Alibi		✓	✓		✓	✓	✓
Aix360	✓		✓				✓
Dalex	✓		✓	✓	✓		
Dice						✓	

Table 1. Explainers by library.

work with any ML model and is not limited to a particular form of explanation and representation. An essential requirement for LIME is to work with an interpretable representation of the input, like images or bag of words, that is understandable to humans. The output of LIME is a list reflecting the contribution of each feature to the prediction of a data sample. Although LIME is general and flexible, there are some scenarios where simple perturbations are not enough, so there are other similar approaches like Anchor [12] where perturbations variation depends on the dataset.

Anchors [12] also known as scoped rules, attempt to explain individual predictions of a black-box model by finding a decision rule which allows the perturbation of other feature values without affecting the actual prediction. Similar to LIME, the approach taken by this algorithm is based on a perturbation strategy to generate local explanations, but instead of representing them through a surrogate, these explanations are expressed as IF-THEN rules. This makes anchors very easy to interpret. Since the perturbations are produced and evaluating for every specific instance that is being explained, the internal structure of the model is never referenced; and thus, this algorithm is completely model-agnostic. Furthermore, the rules generated are reusable since through the coverage measure, they state to what other types of instances such rules apply in the perturbation space. This algorithm was developed by the same researchers that proposed LIME.

Shapley Additive Explanations (SHAP) [8] is an explainer method based on Shapley values and local surrogates (LIME). Shapley values, a method from coalitional game theory, aims to fairly distribute the total value among the different features of an instance. The value to be distributed is the outcome given by the model, which in the case of a classification problem is the predicted probability. SHAP developers take this concept one step further by representing Shapley values as an additive linear model, bringing in aspects from LIME. There are two different approaches: KernelSHAP and TreeSHAP. The first one refers to a kernel-based estimation for Shapley values calculation. However, computing Shapley values is computationally expensive, so the researchers developed TreeSHAP, which is a faster alternative to KernelSHAP but only applicable to tree-based models such as random forests. SHAP can also provide a global explanation of the model by calculating all the Shapley values for each instance but depending on the approach and the given data, this calculation could be too slow.

Partial Dependence Plots (PDPs) [6] show the marginal effect one or two features have on the predicted outcome of a model. PDPs explain the relationship between a feature and the target, which could be linear or more complex. One of the main advantages of these plots is their interpretability. In simple words, PDPs represent the average predictions of all instances for a specific feature. However, this function represents how much such a feature influences the outcome only when there is no correlation with other features. This is one of the main downsides of PDPs since they may lead to erroneous interpretations. The plots should show the density of instances along the feature axis, as areas of the graph with lower density are not as reliable as other areas where the instances concentrate.

Accumulated Local Effects (ALE) plots [2] have the same purpose as PDPs: to describe how one or two specific features affect the prediction of the model on average. Nevertheless, the difference is that ALE plots are unbiased, meaning that they are still reliable when features are correlated. They are also faster to compute since they scale linearly. One of the key characteristics of ALE is that the calculation of the average effect is separated through intervals of that feature. The mathematical background and implementation of ALE plots are far more complicated than partial dependence, but most PD plots shortcomings are covered by using ALE plots when working with correlated features.

Counterfactual explanations [13] are one of the best local methods when the target audience of the explanations is the customers or end-users that need to have a better understanding of a specific prediction to know how to alter the input to obtain a different result. The idea behind it is to provide a new instance that is as similar as possible to the original instance, but whose prediction differs from the original. There are several aspects to be considered when choosing counterfactuals as the explanation method. First, counterfactuals instances should vary as few features as possible. Second, a counterfactual should have feature values that are likely to be present in a real instance. Otherwise, the counterfactual should be discarded. Lastly, whenever possible, multiple counterfactuals that differ from each other should be provided. Counterfactuals have many advantages; they are easy to implement and do not require access to the training data. However, their main advantage is that the explanation given is very clear and easy to understand, even for people with little to no background in ML or statistics.

Contrastive Explanation Method (CEM) [4] provides local explanations for black-box classification models in terms of Pertinent Positives (PP) and Pertinent Negatives (PN). Pertinent positives represent the features that should be minimally and sufficiently present to predict the same class as the original instance. On the other hand, Pertinent Negatives represent the features that should be minimally and sufficiently absent from the original instances to maintain the predicted class. In a certain way, Pertinent Positives can be compared to anchors, while Pertinent Negatives are similar to counterfactuals. According to the authors, the explanations provided by CEM are clear and intuitive since it states that the instance is classified in

Feature	Mean	Std.	Min	25%	50%	75%	90%	Max
Age	26.82	8.50	13.00	20.00	25.00	32.00	37.00	84.00
Num. of sexual partners	2.51	1.64	1.00	2.00	2.00	3.00	4.00	28.00
First sexual intercourse	16.98	2.80	10.00	15.00	17.00	18.00	20.00	32.00
Num. of pregnancies	2.19	1.43	0.00	1.00	2.00	3.00	4.00	11.00
Smokes (y/n)	0.14	0.35	0.00	0.00	0.00	0.00	1.00	1.00
Smokes (years)	1.20	4.06	0.00	0.00	0.00	0.00	3.00	37.00
Hormonal Contraceptives (y/n)	0.69	0.46	0.00	0.00	1.00	1.00	1.00	1.00
Hormonal Contraceptives (years)	1.97	3.60	0.00	0.00	0.25	2.00	7.00	30.00
Intrauterine device (y/n)	0.10	0.30	0.00	0.00	0.00	0.00	0.00	1.00
Intrauterine device (years)	0.44	1.81	0.00	0.00	0.00	0.00	0.00	19.00
Sexually transmitted disease (y/n)	0.09	0.29	0.00	0.00	0.00	0.00	0.00	1.00
Num. STDs	0.16	0.53	0.00	0.00	0.00	0.00	0.00	4.00
Num. STD diagnoses	0.09	0.30	0.00	0.00	0.00	0.00	0.00	3.00
Biopsy	0.06	0.25	0.00	0.00	0.00	0.00	0.00	1.00

Table 2. Basic statistical description of the data set used to evaluate the libraries.

the predicted class because some specific features are present and because some specific features are absent.

3 Methodology

In order to evaluate the XAI libraries we have defined different variables that let us compare the features of each library. These variables are analysed by implementing the same use case consisting of explaining a prediction model. The resulting quantitative analysis of the libraries is presented in Section 4, whereas a qualitative evaluation focusing on the visualization of the methods is included in Section 5.

The review criteria used for the evaluation of the libraries was focused on the following variables:

Documentation and usability. Is the documentation well-structured and self-explanatory? Good documentation should be complemented with usage examples which makes the library easier to use.

Interpretability metrics. Refers to the availability of metrics such as accuracy, recall, ROC/AUC values, mean squared error, etc. These metrics allow users to evaluate the performance of a model.

Available explainers such as LIME, SHAP Kernel/Tree, Counterfactuals, Anchors, etc. Section 2 describes several basic explainers.

Analysis and description capabilities of the training data: refers to the availability of tools that allow a better interpretation of data itself such as marginal and scatter plots, data imbalances, etc.

Interactivity, meaning the user is able to dive deeper into the explanation that is outputted by looking into certain features or other aspects more thoroughly.

Personalization. Refers to the capability of providing different explanations according to the user's requirements.

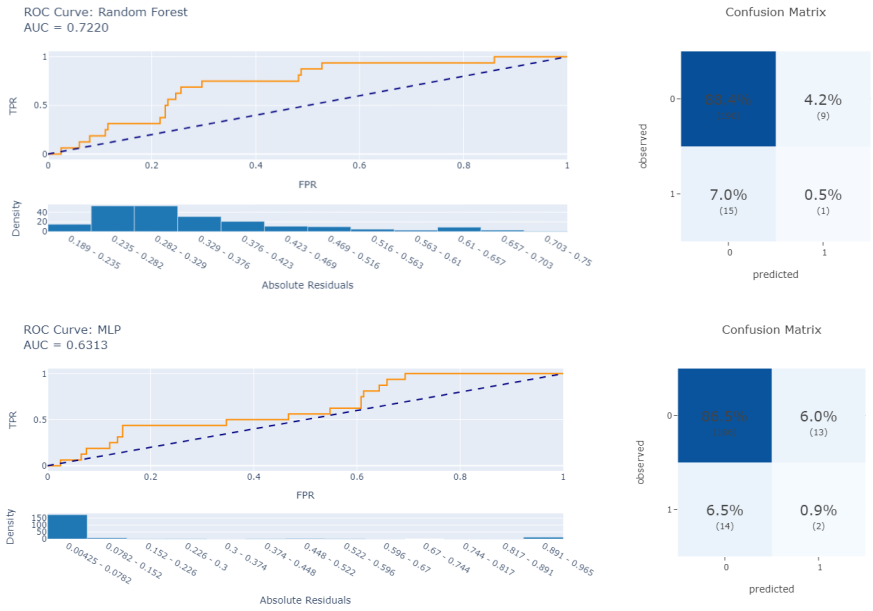

Fig. 1. ROC/AUC and confusion matrix (with a 0.25 threshold) for the RF and MLP models (thresholds: .5 and .24). Plot from InterpretML.

Dependencies. Development language/environment and requirements (if any). Use of other methods from libraries such as tensorflow, sklearn, and others. We also take into consideration the use of wrapper classes and methods of the original author´s implementation of certain explainers.

The use case consists on explaining the prediction of cervical cancer given by two different models: a random forest (RF) classifier and a multi-layer perceptron (MLP), both with a scikit-learn back-end.

The dataset used to train both models was extracted from the UCI Machine Learning repository [5] and contains 858 instances. In Table 2, several statistical descriptors such as the mean, standard deviation, percentiles, and minimum and maximum values are described. It is critical to highlight the fact that the data set is quite unbalanced, as only 6% of the individuals had cervical cancer.

The RF model was built with 100 estimators and was configured so it would adjust the weights inversely proportional to class frequencies. In this way, it is possible to mitigate data imbalances moderately. However, this approach cannot be done when building an MLP, which affected the performance of the model considerably. Our MLP was built with two hidden layers, 100 neurons for the first and 50 neurons for the second. The selected optimization algorithm was Adam. In Figure 1, the confusion matrices and ROC/AUC values for both the random forest and MLP models are respectively shown. The random forest had an accuracy of 88.8%, a precision of 10%, and a recall of 6.2%. On the other

	Interpret	Dice	ALIBI	Aix360	Dalex
Documentation and usability	Very good	Good	Very good	Very good	Good
Metrics	ROC/AUC	No	Linearity measure and trust scores	Faithfulness and monotonicity	F1, accu, prec, recall, ROC/AUC, R^2, MAD
Explainers	3	1	5	3	4
Analysis	Yes	No	No	Yes	No
Interactivity	Yes	No	No	No	Yes
Personalization	No	No	No	No	No
Dependencies	Python 3.6+	Python 3+	Python 3.6+	Python 3.6+	Python 3.6+

Table 3. Features of two instances predicted as positive (instance A) and negative (instance B) by the model for cancer.

hand, the MLP model had an accuracy of 87.4%, a precision of 13.3%, and a recall of 12.5%. It is shown that both models have a considerable rate of false negatives which may be something to take into account because of the sensitive nature of this particular problem. Although the performance of both classifiers is far from perfect, it is important to consider that building a classifier that aims to predict cancer is not an easy task, especially considering the imbalances of the target classes and the small size of the data set.

4 Quantitative analysis of the XAI Libraries

This section describes the XAI libraries being analyzed: InterpretML, ALIBI, Aix360, Dales and Dice. Their corresponding analyses according to the features described in the previous section are included in Table 3.

InterpretML . InterpretML is one of the most popular XAI libraries. It offers state-of-the-art explanations for black-box models both locally and globally. It implements a dashboard that makes the communication process between the end-users and the program more interactive, allowing them to have a better understanding of the explanation.

Dice. Dice, whose name comes from Diverse Counterfactual Explanations, uniquely focuses on counterfactual generation. Three different approaches can be taken when using dice in order to find counterfactuals: using random sampling, k-d trees, or genetic algorithms. Its simplicity of use makes Dice a great candidate when the only explanation needed is various counterfactuals.

ALIBI . Alibi provides local and global explanation methods for classification and regression problems for both with and black-box models. It is a broad

Fig. 2. Screenshot of the Arena dashboard.

library with many different explainers. One of the strengths of this library is that some explainers are compatible with Tensorflow models, such as CEM and counterfactuals, thus increasing its versatility.

Aix360. Aix360 is a multipurpose library that provides some of the most up-to-date explainers available. Besides implementing the widely accepted LIME and SHAP methods, algorithms like Protodash (Gurumoorthy et al., 2019) and CEM with Monotonic Attribute Functions (Luss et al., 2019) show some of the latest, local explainers available. This library also provides global explainers methods such as Generalized Linear Rule Models (Wei et al., 2019) and performance metrics of the model.

Dalex. Dalex is a multipurpose library that focuses on model-agnostic explanations for black-box models. The core methodology behind it is to create a wrapper around the given model that can later be explained through a variety of local and global explainers. This library implements well-known explainers such as LIME, SHAP, and ALE, and also allows measuring the fairness of the model. It provides plenty of different performance metrics according to the given model. Dalex is complemented by the *Arena* visual dashboard, that allows interactive exploration and personalization of the explanation. Figure 2 shows an screenshot of Arena [2].

5 Qualitative evaluation of the XAI methods

This section presents a descriptive evaluation of the XAI methods provided by the libraries, focusing on the visualization of the explanations.

In order to grasp a general idea of the inner mechanics of the models, using **SHAP** as a global explanation method is typically a good choice, although it

[2] https://arena.drwhy.ai/docs/

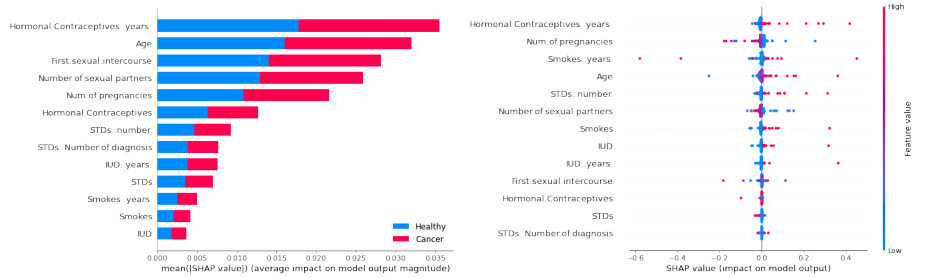

Fig. 3. Average SHAP values for the RF (left) and MLP (right) models. Plot from Dalex.

is not always possible due to its high computational cost. The results obtained for our use case are shown in Figure 3. The features that impact the prediction the most on average for the random forest model are the number of years using hormonal contraceptives, the age of the individual, and the age of their first sexual intercourse. The years of smoking barely contribute to the predictions of the model on average. On the other hand, the SHAP summary plot for the MLP model, which may be somewhat harder to understand, still gives the major contribution to the hormonal contraceptive feature, but then comes the number of pregnancies and the years of smoking. Something interesting about this plot is that the years of smoking contribute both negatively and positively in different situations when the instance values are high, which might indicate that the model is not properly calibrated.

The **partial dependence plots** are also useful when globally examining the behavior of a single feature. In Figure 4, the respective plots of the random forest and MLP models are shown for the feature of years using hormonal contraceptives. Although the average impact on the random forest model is higher, the interpretation is the same for both plots; the more years using hormonal contraceptives, the greater the average response on the prediction is. However, this last statement is only true when variables are not correlated. Furthermore, the density indicates that most instances focus on a range between 0 and 1.88 years which makes the resulting graphs less reliable as the value of this feature increases.

An unbiased alternative method that does consider correlations is **ALE**. The ALE plot for this same feature is shown in Figure 5. In this plot, the average response for the target classes is represented. Since this is a binary classification problem, the graph is perfectly symmetrical. The function behaves similarly to the ones in the PDPs but there seems to be a peak in the average response around 4 years of use, then the response diminishes and stays constant until it reaches around 8 years. Then the average response abruptly increments indicating a bigger impact on the prediction. Although ALE plots are an excellent way to cope with the shortcomings of PDPs regarding correlation, the reliability related to the density of instances is still the same.

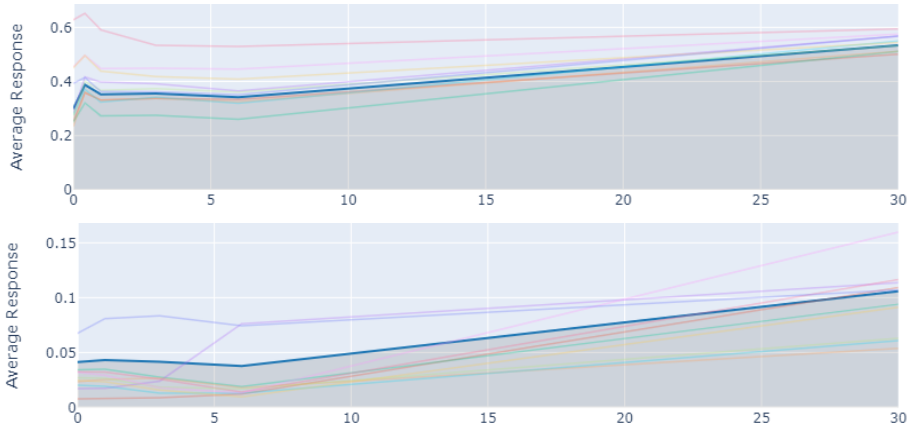

Fig. 4. Partial dependence plot of the years using hormonal contraceptives feature of the RF (top) and MLP (bottom) models. Plot from InterpretML.

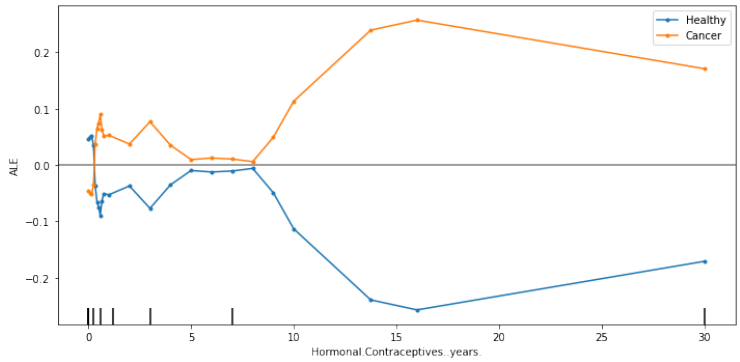

Fig. 5. ALE plot of the years using hormonal contraceptives feature of the random forest model. Plot from Alibi.

When the aim is to explain individual predictions, **LIME** is one of the most used methods. It perturbs the dataset to get predictions for new, proximate samples that allow adjusting the weighting and training of an interpretable, linear model. This interpretable model provides a local explanation because its training is based on the proximity of the generated data points to the original instance. In Figure 6, a specific instance A is explained using LIME on the random forest model. The attributes of instance A, that obtains a positive prediction, are presented in Table 4. The plot shows that the features that affect the prediction the most around the given instance are hormonal contraceptives and STD-related ones. Other features, such as years of smoking and the number of pregnancies are considerably less impacting, even though they have high values in comparison with the average. This interpretation may represent properly the behavior

Features	Instance A (+)	Instance B (-)
Age	52	25
Number.of.sexual.partner	5	2
First.sexual.intercourse	16	18
Num.of.pregnancies	4	2
Smokes	1	10
Smokes..years.	37	0
Hormonal.Contraceptives	1	1
Hormonal.Contraceptives..years.	3	0.25
IUD	0	0
IUD..years.	0	0
STDs	0	0
STDs..number.	0	0
STDs..Number.of.diagnosis	0	0

Table 4. Summary of the main characteristics of the reviewed XAI Libraries.

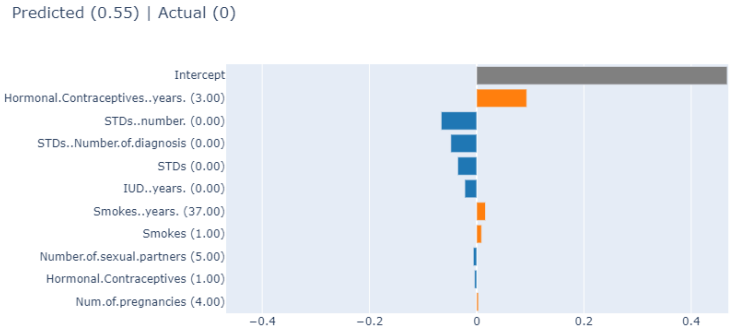

Fig. 6. LIME plot of an instance predicted positively by the model. Plot from InterpretML.

of the model locally around the given instance. However, it does not necessarily represent the global behavior of the model. Another aspect to take into consideration with LIME is that the explanations may vary considerably for two similar instances [1]. This leads to explanations that are not as trustful as needed in some scenarios.

A similar approach to LIME is taken when using **Anchors**. Anchors provide conditions that are locally sufficient to determine a prediction with a certain degree of confidence. Let us look at Instance in Table 4. This instance was predicted as negative for cancer. Using anchors we obtain the following conditional rule:

```
Anchor:    Age <= 31.00 AND
           STDs..number. <= 0.00
Precision: 0.97
```

```
Coverage:  0.69
```

The anchor given is that when the age is less or equal to 31 and the individual has not had any STD, the model classifies the individual as healthy with a precision of 97% and the coverage, representing the extent of the area of the perturbation space to which this rule applies, is rather high with 69%. The simplicity of anchor makes them excellent to obtain local explanations that are easy to interpret. However, the given rule may be too complicated or have low precision and coverage in certain cases. For example, when trying to find an anchor for instance A described in Table 4, this is the result:

```
Anchor:    Hormonal.Contraceptives..years.> 2.00 AND
           15.00 < First.sexual.intercourse < 17.00 AND
           Age > 31.00 AND
           Num.of.pregnancies > 2.00 AND
           Smokes..years. >= 0.00 AND
           Hormonal.Contraceptives >= 0.00
Precision: 0.37
Coverage:  0.01
```

As it is seen, the anchor obtained is way too complex since it involves too many features, and both precision and coverage values are very low. This leads to a rule that may not be very accurate and is difficult to trust. Nevertheless, depending on the behavior of the model and the nature of the problem being solved, anchors represent a good alternative to obtain local explanations.

If the focus is to provide contrastive, concise, and easy-to-interpret individual explanations, **counterfactuals** are one the best choices. The goal is to get instances that are very similar to the original instance, but that would be labeled as a different class by our model. Using again the individual predicted as positive for cancer by the model from table 4, we restrict the features to vary to years of smoking, the number of pregnancies, years using hormonal contraceptives, and the number of sexual partners. The counterfactuals generated are shown in Table 5; only the indicated features are included, the rest remains the same. All the counterfactuals generated show a considerable decrease in the years of smoking value, but there are interesting combinations of features. For example, if the individual had had only 1 pregnancy instead of 5, smoked for 24.7 years instead of 37, the classification would have changed. The instances generated give insight about what could have been done differently with the purpose of being predicted by the model as a healthy individual. Although it is impossible for individuals of this data set to change the characteristics they already have, counterfactuals work perfectly when features can be modified towards a certain goal, such as being approved for a bank loan.

6 Conclusions

This review of the XAI libraries allowed us to have a better understanding of some of the most popular and up-to-date explainers that machine learning and

Counterfactual	Sexual partners	Pregnancies	Smokes (years)	Contraceptives (years)
1	-	-	5.3	0.7
2	1	-	8.6	-
3	-	1	24.7	-

Table 5. Counterfactuals generated using DICE. The cells containing a hyphen represent no change from the original instance.

data scientists use to explain black-box models. Although all the libraries reviewed had their pros and cons, some of them proved to be highly versatile and interactive, making the process of obtaining good explanations considerably easier. To conclude this paper, we provide some subjective opinions of each library regarding its usability, variety, interactivity, and other characteristics. If we had to rank the libraries, InterpretML would probably get first place. Even though is not the most extensive library, its usability and neat interfaces make it the number one choice for explainability. Interpret is very easy to use as most of its explainers barely require a single function call specific to the explainer used. The explanations generated are shown in the dashboard, which is an interactive interface that allows switching the visualization depending on the attribute that is emphasized, and even shows different explanations for the same model. This makes Interpret a very versatile tool if what we need is to obtain various explanations and compare them in a way we can choose the one that better fits our needs. Additionally, its documentation is well structured and complemented with several examples. It is a library that a person with little experience in machine learning would be able to use properly in a short time. However, this library only provides LIME and SHAP as local explainers, and partial dependence plots, which does not provide the same reliability as ALE plots. If Interpret widened its explainer repertoire, it would undoubtedly be the best option for machine learning explainability. Curiously, Interpret developers have also developed Dice, which is a separate library that uniquely focuses on counterfactual generation. Although Dice is considerably different from the rest of the libraries reviewed in this paper, it proved to be a solid option to obtain counterfactual explanations. In fact, the algorithm configuration is much more straightforward and intuitive than the one in Alibi. This library also outputs the counterfactuals in an easy-to-understand fashion by using dataframes. Generally speaking, it is easy to use, and the examples provided in the documentation are illustrative and completely model-agnostic, in contrast to Alibi. In most aspects, Dice is considerably better than the approach offered in Alibi as it allows generating counterfactuals easily and outputting them in an interpretable way. Unfortunately, a simple and interactive visualization of explanations is not available in most of the XAI Libraries. Moreover, for counterfactual generation and CEM from Alibi or Aix360, the explanations are given in a low-level format that is hard to read and comprehend. Consequently, the programmer must process this data in order to convert it to a more readable format. This is one of the main

issues of both libraries since explainers do not provide a high-level abstraction of the output so end-users can easily understand the explanations. Although Alibi is the most extensive library out of all reviewed ones, the way the explanations are outputted is somewhat of a letdown. Furthermore, many of the usage examples given are heavily oriented to Tensorflow models, which is a disadvantage when the model to be explained has a different backend. Despite the fact that the documentation is very specific and illustrative of the concepts behind each of the explainers available, for users without a deep background in machine learning and interpretability, using this library may prove to be difficult. On the good side, Alibi has a wide variety of explainers and is the only reviewed library that offers explanations through anchors. However, it does not include LIME. On the other hand, Aix360 is not as complete as Alibi regarding basic explainers, but it includes many other innovative model and data explanation methods such as Protodash and Profweight that may be worth diving deeper into. There are also other global explanation methods such as Boolean Decision Rules via Column Generation [3] and Generalized Linear Rule Models [14] that are not available at any other library than Aix360. Moreover, its documentation is well developed and there are many tutorials available on the official website. However, the implementation of basic explainers such as LIME, SHAP, and CEM does not offer any advantages over other libraries that also implement them. Lastly, we have Dalex, which is not so different from the previous libraries described. One of the few reasons to use it over Interpret is that it provides ALE plots, while only PDPs are available in Interpret. It does not have contrastive methods such as counterfactuals and CEM but it does provide tools for data analysis and feature importance methods. The documentation is appropriately organized but some of the methods are outdated, specifically the ones for the SHAP plotting. The aspect that differentiates Dalex from the other multipurpose libraries reviewed is that it does provide interactivity through the Dalex Arena. In conclusion, choosing one of these libraries over the others depends on the specific needs and preferences of the person who will be using them since there is considerable overlapping between them.

We conclude that one of the greatest downfalls of the XAI libraries currently available is the lack of interactivity and personalization of the explanations. Excepting InterpretML and Dalex, which somewhat allow a simple interaction between the user who receives the explanations and the program, none of the libraries reviewed provide any form of user interaction nor personalization.

The idea behind the iSee project is to provide personalized explanations that suit the needs of the person receiving them, by analyzing user interactions using a case-based reasoning system. In this way, it will be possible to merge the already existing explainability methods with a user-oriented approach that aims to improve the machine learning interpretability field.

References

1. Alvarez-Melis, D., Jaakkola, T.S.: On the robustness of interpretability methods (2018)
2. Apley, D.W., Zhu, J.: Visualizing the effects of predictor variables in black box supervised learning models (2019)
3. Dash, S., Günlük, O., Wei, D.: Boolean decision rules via column generation (2020)
4. Dhurandhar, A., Chen, P.Y., Luss, R., Tu, C.C., Ting, P., Shanmugam, K., Das, P.: Explanations based on the missing: Towards contrastive explanations with pertinent negatives (2018)
5. Dua, D., Graff, C.: UCI machine learning repository (2017), http://archive.ics.uci.edu/ml
6. Friedman, J.H.: Greedy function approximation: A gradient boosting machine. The Annals of Statistics **29**(5), 1189 – 1232 (2001). https://doi.org/10.1214/aos/1013203451
7. Lipton, Z.C.: The mythos of model interpretability. Commun. ACM **61**(10), 36–43 (2018). https://doi.org/10.1145/3233231
8. Lundberg, S., Lee, S.I.: A unified approach to interpreting model predictions (2017)
9. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. CoRR **abs/1706.07269** (2017), http://arxiv.org/abs/1706.07269
10. Molnar, C.: Interpretable Machine Learning (2019), https://christophm.github.io/interpretable-ml-book/
11. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?": Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 1135–1144. Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2939672.2939778
12. Ribeiro, M.T., Singh, S., Guestrin, C.: Anchors: High-precision model-agnostic explanations. In: McIlraith, S.A., Weinberger, K.Q. (eds.) Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI-18. pp. 1527–1535. AAAI Press (2018)
13. Verma, S., Dickerson, J., Hines, K.: Counterfactual explanations for machine learning: A review (2020)
14. Wei, D., Dash, S., Gao, T., Günlük, O.: Generalized linear rule models (2019)
15. Weld, D.S., Bansal, G.: The challenge of crafting intelligible intelligence. Commun. ACM **62**(6), 70–79 (2019). https://doi.org/10.1145/3282486

A Appendix:Quantitative analysis of XAI libraries

Documentation and usability	The documentation is well-structured and explanatory. Usage examples are provided in a simple fashion so the user is able to begin using the library very quickly. This library is very intuitive and using it should not arise any issues for less-experienced users.
Metrics	ROC/AUC values.
Explainers	3
Analysis	Yes. It provides marginal plots and class histograms.
Interactivity	It has a dashboard feature that allows the end-user to further inquire into different features and compare different explanations of the same instance.
Personalization	Not available
Dependencies	Python 3.6+. For the LIME and SHAP explainers, wrapper classes are used based on the original implementation developed by [11] and [8], respectively.

Table 6. Analysis of InterpretML.

Documentation and usability	The documentation is straightforward and provides various examples. It is very simple as this library uniquely relies on counterfactual generation.
Metrics	Not available.
Explainers	1
Analysis	Not available.
Interactivity	This library does not provide interactivity, but the data is presented in an easy-to-interpret format.
Personalization	Not available.
Dependencies	Python 3+. It does not use other external interpretability libraries. However, depending on the backend of the model, it may rely on Tensorflow and Pytorch.

Table 7. Analysis of Dice.

Documentation and usability	The documentation is very extensive and educational. Not only does it explain how to use the methods, but gives a mathematical background for each explainer. However, the examples provided for some explainers only cover the explanation of models with a Tensorflow backend, which may cause difficulties to users who are not experienced in this environment.
Metrics	Linearity measure and trust scores.
Explainers	5
Analysis	Not available.
Interactivity	This library is not interactive. The process is finished once the explanation is outputted. In fact, most explanations are given in a low-level fashion as raw data that the user may need to convert to a more interpretable format.
Personalization	Not available.
Dependencies	Python 3.6+. This library is heavily based on tensorflow. For the SHAP explainer, it uses the original implementation of the author [8].

Table 8. Analysis of ALIBI.

Documentation and usability	The documentation is explicative and extensive. It provides many usage examples with different data sets that make the library easy to use. The Aix360 website offers interactive tutorials as complementary guidance for its use.
Metrics	Faithfulness and monotonicity. Faithfulness refers to the correlation between the feature importance assigned by the interpretability algorithm and the effect of features on model accuracy. On the other hand, monotonicity tests whether model accuracy increases as features are added in order of their importance.
Explainers	3
Analysis	Yes. Particularly, the Protodash algorithm is able to find prototypes that help summarizing the data set.
Interactivity	This library does not provide the interactivity feature. Explanations are outputted to the users in the format of graphics or plain data, and the is no further interaction between the user and program.
Personalization	Not available. However, the importance of personalization of the explanations is referenced in the official website throughout the interactive demo. It shows that different users look for different kinds of explanations. This is one of the main ideas behind the iSee project, to provide the users with the explanations that best suit their needs in an interactive fashion.
Dependencies	Python 3.6+. The implementation of the original authors is used for the LIME and SHAP explainers [11][8]

Table 9. Analysis of Aix360.

Documentation and usability	The documentation is good and plenty of examples are provided. Other complementary resources such as tutorials are provided as well. However, it may be hard to find the exact usage illustration for a specific explainer in a notebook since they are organized by data sets.
Metrics	There are many different metrics provided depending on the nature of the problem. For classification, F1 score, accuracy, recall, precision, specificity, and ROC/AUC are provided. For regression problems there is mean squared error, R squared, and median absolute deviation.
Explainers	4
Analysis	Not available.
Interactivity	The Dalex Arena allows the user to easily compare different explanations for the same problem, and even different models.
Personalization	Not available.
Dependencies	Python 3.6+. For the LIME and SHAP explainers, wrapper classes are used based on the original implementations [11][8].

Table 10. Analysis of Dalex.