Explainable Debugger for Black-box Machine Learning Models

1st Peyman Rasouli
Department of Informatics
University of Oslo
Oslo, Norway
peymanra@ifi.uio.no

2nd Ingrid Chieh Yu
Department of Informatics
University of Oslo
Oslo, Norway
ingridcy@ifi.uio.no

***Abstract*—The research around developing methods for debugging and refining Machine Learning (ML) models is still in its infancy. We believe employing tailored tools in the development process can help developers in creating more trustworthy and reliable models. This is particularly essential for creating black-box models such as deep neural networks and random forests, as their opaque decision-making and complex structure prevent detailed investigations. Although many explanation techniques provide interpretability in terms of predictive features for a mispredicted instance, it would be beneficial for a developer to find a partition of the training data that significantly influences the anomaly. Such responsible partitions can be subjected to data visualization and data engineering in the development phase to improve the model's accuracy. In this paper, we propose a systematic debugging framework for the development of ML models that guides the data engineering process using the model's decision boundary. Our approach finds the influential neighborhood of anomalous data points using observation-level feature importance and explains them via a novel quasi-global explanation technique. It is also equipped with a robust global explanation approach to reveal general trends and expose potential biases in the neighborhoods. We demonstrate the efficacy of the devised framework through several experiments on standard data sets and black-box models and propose various guidelines on how the framework's components can be practically useful from a developer's perspective.**

***Index Terms*—Trustworthy Machine Learning, Black-box Models, Model Debugging, Data Engineering, Explainability, Quasi-Global Explanations**

I. Introduction

Data engineering is an inseparable part of the Machine Learning (ML) development workflow, which highly affects the performance of a resultant model. In this phase, we identify the problems associated with the data set and endeavor to refine them to create a better ML model. Traditional techniques mostly concentrate on the feature space and the model's predictions, but they do not take into account the structure of the decision function [1]. Although we benefit from applying these preprocessing methods, when a developer faces a misprediction, he may be interested in finding the answer to the following questions: 1) what subset of training data most influenced this prediction? 2) what are the important features in the discovered subset that affect the prediction? 3) what are the problems associated with this subset (e.g., bias, class imbalance, etc.)? 4) what is the explanation for the anomaly and its retrieved subset? Answering these questions can help the developer peek into different partitions of the data and perform strategic data cleaning, data gathering, and model augmentation.

For a specific choice of ML model, anomalies can occur due to various factors such as class imbalance, data bias, non-linear data, etc., and the severity of the anomalies depends on the expressiveness of the model. Having tools in the training phase that assist the developer to debug anomalies regarding the model's output can increase the quality and the trustworthiness of the deployed model. This is even more critical for black-box ML approaches like deep neural networks and random forests because their opaque decision-making procedure prevents direct debugging and interpretation [2]. The complexity imposed by such models calls for developing tailored debugging and explanation techniques that can reveal misbehaviors and enhance the understandability of the model's decision-making. Techniques that identify and explain anomalies can provide developers with interactive data analysis to achieve highly accurate ML models.

There are substantial research works around data preprocessing, model validation, and model calibration [1], [3], [4]. A commonality between these approaches is relying on the feature space without considering the decision boundary of the black-box. Consequently, these techniques focus on global data improvement instead of identifying the causes of individual anomalies. The ML development process is an iterative task that involves the black-box model as an indispensable part. Therefore, debugging techniques must take into account both feature space and decision boundary in order to precisely locate the causal effects of anomalies. There are a variety of local and global explanation methods [5] that provide useful insights about the model's decision-making mechanism. LIME [6] explains the prediction of a particular instance through creating a linear model trained on the locality of the instance where coefficients of the model present the importance of each individual feature. EXPLAN [7] is a model-agnostic rule-based explanation method for tabular data that justifies the output of a black-box model using IF-THEN statements. TreeExplainer [8] is an explanation approach for tree-based models based on coalitional game theory, which calculates the exact contribution of features in the model's prediction using Shapley values. Partial Dependence Plot (PDP) [9] and Accumulated Local Effects (ALE) [10] are global explanation methods that reveal the average relationship of features with the predicted outcome of a black-box model. Although post-hoc explanation techniques are successful in providing transparency for black-box models, they only justify the models' predictions. In contrast, having a method during the development/training phase that can help us refine the data with respect to the decision function is useful for creating a model that performs well in the deployment/testing phase.

In this paper, we propose a general framework that helps the developers through the ML development pipeline. The applications range from data cleaning, bias reduction, mislabeled data detection, and feature engineering. Although there are tools and algorithms to perform data preprocessing tasks [1], no work exploits the information derived from decision function structure into these operations to the best of our knowledge. This work is the first attempt to bring the utility of explanation methods into ML models' development. From our perspective, creating a black-box machine learning model is an iterative development process where the output of the created black-box can be fed into the data engineering pipeline for further refinement. In addition to the feature values, we incorporate knowledge about the decision function into the development process, improving the construction of a black-box model with the desired level of accuracy. By focusing on the mispredicted samples (anomalies) in each iteration of the development process, the proposed framework identifies influential samples in the training data that have contributed to each anomalous instance. Moreover, local and global explanations are provided to enhance the understandability of the occurred misbehaviors. The main contributions of this work are as follows:

1) Creating a nearest neighbor model based on feature contributions for finding the most influential samples in the training data for an anomalous instance.
2) Devising a quasi-global explanation method based on genetic algorithm that selects representative samples in the influential neighborhood of an anomaly and provides a set of local rule-based explanations.
3) Explaining the influential neighborhood of an anomalous instance globally to identify the important features for the neighborhood and to reveal potential biases.

For each contribution, we provide several guidelines for the developers that can assist in finding the potential benefits of the proposed techniques. This work mainly focuses on the theoretical aspects of the proposed framework and leaves the practical investigation for future work. The rest of the paper is organized as follows. Proposed methodology is described in Section II. Section III reports validation and analysis results of the devised framework. Section IV concludes the paper and states future works.

II. Proposed Debugging Framework

In the following, we present our debugging framework that is designed to identify and explain the influential neighborhood of anomalies. The framework consists of three components:

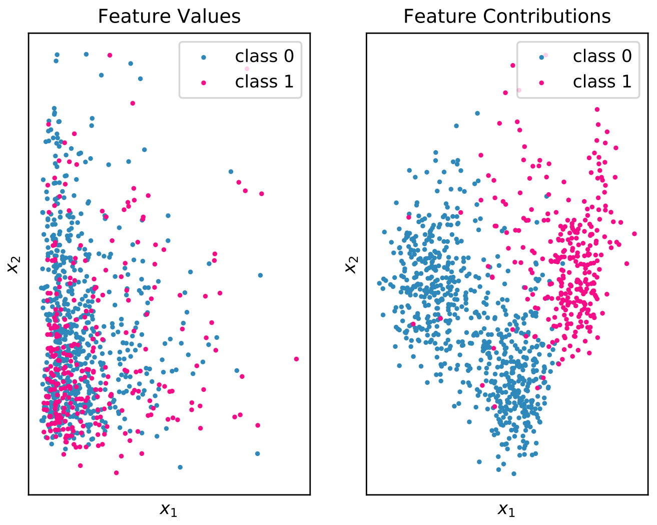

Fig. 1: Visualization of data points in different perspectives: feature values vs. feature contributions.

neighborhood model construction, quasi-global explanation of anomalies, and global explanation of influential neighborhoods. Although we have created the method for refining the black-box model during the training phase, the developer can adopt the methodology in the deployment phase to warn potential anomalous instances. A complete implementation of the methodology along with the experiments is available at: https://github.com/peymanrasouli/XdebugML.

A. *Influential Neighborhood of Anomalies*

Anomalies are the result of blind spots, regions in the decision function that cause model's misbehavior. Identifying the most influential samples in the prediction of an anomalous instance is considered as the first and essential step of the debugger framework. Although creating a neighborhood using samples with similar feature values to the anomaly seems a reasonable idea, it is noteworthy that a diverse set of training samples form the non-linear decision boundaries of a ML model. Therefore, finding a set of samples in the training data which are contributing to the prediction of an anomaly merely through feature value comparison is not practical.

In this section we introduce an algorithm for finding influential neighborhood for an anomalous instance using observation-level feature importance/contributions. We employ feature contributions because they provide useful information about the structure of the black-box decision function. We can consider feature values and feature contributions as different perspectives of the data, where the former is merely about the distribution of the data, while the latter is the result of the created decision boundary of a particular black-box model. We illustrate this distinction in Fig. 1 by plotting the data points of *German credit* [1] data set in a 2D space. We used Principal Component Analysis (PCA) [11] to reduce the

[1]Data set is available at: https://archive.ics.uci.edu/ml/datasets/

feature values and feature contributions into two dimensions, denoted by x_1 and x_2. It can be seen that feature contributions are highly distinguishable regarding the model's prediction. This allows us to find samples that are contributing to a specific region of the decision space. Consequently, this is beneficial for recognizing the neighborhood of a blind spot for further investigation.

Algorithm 1 Neighborhood Model Construction

Input: f: black-box model, X_{train}: training data, K: number of neighborhood samples

Output: N_{model}: neighborhood model, C: feature contributions

```
1:  function $\Omega(f, X_{train}, K)$
2:      procedure ContributionExtraction($f, X_{train}$)
3:          $C \leftarrow \{\}$
4:          for all $x \in X_{train}$ do
5:              $l \leftarrow f(x)$
6:              $c \leftarrow ShapleySamplingValues(f, x)$
7:              $C \leftarrow C \cup c_l$
8:          return $C$
9:      procedure NeighborhoodModel($C, K$)
10:         $N_{model} \leftarrow KNN\text{-}Constructor(C, K)$
11:         return $N_{model}$
12:     return $N_{model}, C$
```

To achieve the influential samples of anomalies we create a neighborhood model based on the training data. Given a tabular data set $D = (X, Y)$, we split the data into train $D_{train} \in D$ and test set $D_{test} \in D$. A black-box classifier $f : X \rightarrow Y$ using D_{train} is then created that maps inputs X to the labels Y. We call an instance $(x, y) \in D_{train}$ as an anomaly if $f(x) \neq y$. The problem is to find data points in the training data, D_{train}, that are contributing to the prediction of x. These samples form a neighborhood $N \subset D_{train}$ that is responsible for the prediction $f(x)$. A simple way to evaluate the influence of N is by perturbing the label of its members, creating a new training data $\hat{D}_{train}$, and training a new black-box classifier $\hat{f}$ (same class as f with the same seed and hyper-parameters) using $\hat{D}_{train}$. An influential neighborhood should normally change the prediction of the anomalous instance in the new classifier, i.e., $\hat{f}(x) = y$. We propose a function Ω for creating a neighborhood model N_{model} based on feature contributions, defined in Eq. (1):

$$N_{model}, C = \Omega(f, X_{train}, K) \tag{1}$$

where f is the black-box classifier, X_{train} refers to the inputs in the training data, and K is the number of desired neighborhood samples. The implementation of function Ω, which returns a neighborhood model N_{model} and feature contributions of the training data C, is described in Algorithm 1. Our devised algorithm is model-agnostic, so it can be applied to any tabular black-box classifier regardless of its internal structure and mechanism.

Algorithm 1 consists of two steps for creating the neighborhood model that are explained in the following. We reconstruct N_{model} after each cycle of the development process, as modifying the data/model is likely to change the feature contributions. Further, we introduce computationally efficient methods for constructing the neighborhood model with a time complexity comparable to the black-box model construction.

In ContributionExtraction step, for every instance in the training data, observation-level feature contributions are extracted. There are several ways to achieve feature importance such as Shaply sampling values (aka. IME) [12], local explanation methods (e.g., LIME [6] and TreeExplainer [8]), and TreeInterpreter [13]. For tree-based models, TreeExplainer or TreeIntepreter can provide contribution values more efficiently. It is also possible to create a tree-based surrogate model for the black-box and to use a tree-based contribution extractor. However, this may lead to a representation gap between the black-box and the surrogate model. IME is a method based on fundamental concepts from coalitional game theory that measures the contribution of individual feature values in the prediction of a ML model. More precisely, feature values of a data point interact together to cause a change in the model's prediction with respect to the model's expected output (average prediction of the training data); here, IME distributes this change in the prediction among the features in a fair manner that describes the contribution of each feature in the prediction of the instance. The contribution values have either positive or negative sign which indicate their contributions towards increasing or decreasing the model's output, respectively. IME is a model-agnostic approach, therefore it can be applied to any ML model. We use IME for this work and leave the exploration of other contribution extraction techniques for the future work. Shapley sampling values is applied on black-box f and the training data X_{train} to generate feature contributions. Given an instance $x \in X_{train}$, it produces contribution values c of every feature with respect to every class in the data. Eventually, the contribution vector associated to the predicted label by f, denoted by c_l, for $\forall x \in X_{train}$ is returned as matrix C.

In the second step, NeighborhoodModel, a nearest neighbor model [14], denoted by N_{model}, using the feature contribution matrix C is created. Given the contribution vector of a particular instance x, the model outputs the indices of its influential samples in the training data, i.e., $I = N_{model}(x)$. The only required hyper-parameter is the number of neighborhood samples that is determined by K. The selection of K depends on the size of the training data, as larger data sets demand larger values of K, which results in creating a more influential and reliable neighborhood. We study the impact of different values of K in future work.

B. Quasi-Global Explanation of Anomalies

The created neighborhood model, i.e., N_{model}, provides various analysis opportunities for resolving the anomalies and improving the model's development. Local explanation of the adjacent samples can help the domain expert to understand the behavior of the model in the neighborhood of the anomalous

TABLE I: Local explanation of the representative set of the anomalous instance $\hat{x}$.

y	$f(x)$	Explanation Rule
>50K	**<=50K**	**capital-gain<=0 $\wedge$ capital-loss<=0 $\wedge$ 40<age<=53 $\wedge$ relationship=Husband $\wedge$ education=Some-college $\wedge$ hours-per-week<=42**
>50K	<=50K	44<age<=59 $\wedge$ hours-per-week<=47 $\wedge$ capital-gain<=0 $\wedge$ relationship=Wife $\wedge$ education=HS-grad $\wedge$ capital-loss<=0
>50K	<=50K	capital-gain<=4945 $\wedge$ age>44 $\wedge$ relationship=Husband $\wedge$ education=HS-grad $\wedge$ capital-loss<=904 $\wedge$ hours-per-week<=41
>50K	<=50K	capital-gain<=0 $\wedge$ age<=48 $\wedge$ hours-per-week<=43 $\wedge$ relationship=Husband $\wedge$ capital-loss<=0 $\wedge$ education=HS-grad
>50K	<=50K	capital-gain<=2105 $\wedge$ relationship=Husband $\wedge$ capital-loss<=0 $\wedge$ education=HS-grad
>50K	<=50K	hours-per-week<=46 $\wedge$ capital-gain<=0 $\wedge$ age<=49 $\wedge$ capital-loss<=0 $\wedge$ relationship=Wife $\wedge$ education=HS-grad
<=50K	<=50K	relationship=Wife $\wedge$ education=HS-grad $\wedge$ occupation=Prof-specialty
<=50K	<=50K	age>36 $\wedge$ capital-gain<=5178 $\wedge$ relationship=Wife
<=50K	<=50K	age>30 $\wedge$ relationship=Husband $\wedge$ education=5th-6th
<=50K	<=50K	capital-gain<=6849 $\wedge$ relationship=Wife $\wedge$ age>30 $\wedge$ education=HS-grad
<=50K	<=50K	capital-gain<=0 $\wedge$ hours-per-week<=43 $\wedge$ capital-loss<=0 $\wedge$ relationship=Husband $\wedge$ education=HS-grad

instance and to get insights about the causes of the anomaly. Even though local explanation of multiple samples can be insightful, the user may not have time to examine a large number of explanations. Therefore, we denote the user's time for investigating local explanations by a budget B and propose an algorithm for picking a set of B representative and diverse samples for local explanation. Our algorithm can be considered as a 'Quasi-Global' explanation technique which generates non-redundant local explanations where their association can provide a global explanation of the influential neighborhood for the domain expert.

1) Example: Imagine a gradient boosting model f that is trained on *Adult* [2] data set. The data set contains demographic, educational, and other information of individuals for classifying their income to over/under 50K dollars. Consider an instance $(\hat{x}, \hat{y})$ that is mispredicted by the model, i.e., $f(\hat{x}) \neq \hat{y}$. Using N_{model}, we identify the influential neighborhood of $\hat{x}$ and generate its quasi-global explanation via the algorithm described in Section II-B2. The explanation listed in Table I consists of IF-THEN rules for the anomalous instance $\hat{x}$. The first row of the table corresponds to the local explanation of $\hat{x}$ and the rest of the rows are the explanations of the representative set. The selected set is diverse, and this diversity is reflected by the contrastive explanations for anomalous $(f(x) \neq y)$ and non-anomalous $(f(x) = y)$ instances. The generated rules are different regarding feature names, feature values, and length because our proposed algorithm takes into account the diversity of feature contributions for instance selection. A developer can observe noticeable differences between the explanations of the anomalous instances and non-anomalous instances. In this example, it can be seen that explanations of the anomalous instances have a longer length than the explanations of the non-anomalous instances. A likely reason is the position of the anomalous instances in the feature space which are close to the decision boundary of the classes. This results in more complicated explanations. We see the explanations of anomalous and non-anomalous instances are distinguishable in some ways. For example, "**hours-per-week**" is repeated frequently in the explanation of the anomalous instances while it rarely appears in the explanation of the non-anomalous instances. Moreover, anomalous instances have a large value for the feature "**age**" while the value is smaller for non-anomalous instances. Given the derived insights, a possible following action is to look into the correlation of the "**hours-per-week**" and "**age**" features for different classes of the data. By having explanations of several representative sets, a developer may notice difficulties of the classifier in learning the decision boundary with respect to particular features (e.g., "**age**"); in this case, doing feature engineering (e.g., creating new features based on available features like combining "**age**" with "**hours-per-week**") is useful to help the classifier to understand the relationships of features more effectively, and eventually making correct decisions.

2) Algorithm: Instead of focusing on feature values of the samples in the neighborhood, we take into account the importance of each feature in the prediction of the black-box for selecting representative samples. Because, different feature values in various samples may have similar contribution to the black-box prediction, thus it is likely that their corresponding local explanation will be similar as well. Thus, we need to pick a set of diverse instances (from the feature contribution perspective) that maximize the overall importance value. The approach for obtaining diversity and importance information are mentioned below as well as Fig. 2 demonstrates a toy example for each step of the procedure.

Given an anomalous instance x, the indices of its influential neighborhood in the training data are retrieved, i.e., $I = N_{model}(x)$. The corresponding contribution matrix of the selected samples is denoted by $\hat{C}$ where $\hat{C} = \{C_i | \forall i \in I\}$. Utilizing $\hat{C}$, we construct two customized weight matrices that are used as importance and diversity information in the instance selection mechanism. Fig. 2(a) depicts an example of a retrieved neighborhood where I is the single vector (left) and $\hat{C}$ is the matrix (right) with features $F = \{F1, .., F6\}$.

The retrieved importance matrix $\hat{C}$ has the shape $|K \times |F||$ where K is the number of neighborhood samples and F

[2]Data set is available at: https://archive.ics.uci.edu/ml/datasets/

	F1	F2	F3	F4	F5	F6
21	0.05	-0.03	0.06	0.01	-0.04	0.00
8	-0.05	-0.04	-0.03	0.01	0.06	0.01
14	-0.04	-0.03	0.01	-0.08	0.02	0.01
3	0.02	-0.15	0.04	0.01	-0.03	0.00
11	0.09	-0.03	-0.10	0.00	-0.06	0.01

(a) $I, \hat{C}$

1st	2nd	3rd
F3	F1	F5
F5	F1	F2
F4	F1	F2
F2	F3	F5
F3	F1	F5

(b) L

F1	F2	F3	F4	F5
0.05	0	0.06	0	0.04
0.05	0.04	0	0	0.06
0.04	0.03	0	0.08	0
0	0.15	0.04	0	0.03
0.09	0	0.10	0	0.06

(c) W

F1	F2	F3	F4	F5
1	0	1	0	1
1	1	0	0	1
1	1	0	1	0
0	1	1	0	1
1	0	1	0	1

(d) Wb

Fig. 2: Construction procedure of the importance and diversity matrices.

refers to the set of features in the data. A contribution vector $c \in \hat{C}$ contains low-importance and high-importance features with negative or positive sign. Since the importance degree of features is independent of their signs, the absolute value of contributions is used in further operations. Obviously, high-importance features are more informative, and subsequently the local explanation is likely to contain them. In order to foster the selection of representative samples based on high-importance features, we create a list L that contains NF, $2 \leq NF \ll |F|$, top features of every neighborhood instance ($\forall c \in \hat{C}$). After this process, the list L may not include the index of all features, because some low-importance features may be neglected for all instances in the neighborhood. Fig. 2(b) illustrates the list L for $NF = 3$ where high-importance features are placed into first to third places, and it can be seen that feature $F6$ is dropped because it is not among the NF high-important features of the neighborhood samples.

We create a new contribution matrix called W with a size of $|K \times MaxInd(\hat{F})|$ where $\hat{F} \subseteq F$ is the set of unique features available in L and $MaxInd(.)$ is the maximum index of the features in $\hat{F}$. The entries of W are then filled by the corresponding values from $\hat{C}$. For every sample $w \in W$ there may be some features that do not have contribution values as we have selected NF high-importance features for the construction of W. In this case, the value 0 is assigned to these features. Fig. 2(c) demonstrates the generated importance matrix W from the the original neighborhood contribution matrix $\hat{C}$ that has $\hat{F} = \{F1, .., F5\}$ features.

To measure the diversity of the selected samples, we create a binary version of W, called Wb, where features with contribution value greater than 0 are set to 1, otherwise to 0. This allows us to measure the variation of each feature. For example, if a feature $F1 \in \hat{F}$ has the same value (either 0,1) for every sample, it indicates that this feature has low variation and is more likely to be present in most of the B local explanations. Fig. 2(d) depicts the created diversity matrix Wb from the importance matrix W.

We refer to W as an indication of the feature importance and Wb as an indication of the feature diversity and employ them simultaneously to pick B diverse, representative samples from the influential neighborhood. We formulate this as an optimization problem and solve it using the Genetic Algorithm (GA) [15]. Compared to simple greedy search algorithms (like Hill Climbing), GA is likely to find a globally optimal solution by avoiding local optima and effectively searching problem space. These properties are useful when dealing with an extensive data set in which anomalous samples have broad influential neighborhoods, leading to ample search space. To this end, we select B instances by maximizing the following fitness function using GA:

$$\Phi(W, Wb, R) = \underset{R \subset \{1..K\}, \forall j \in \hat{F}}{\mathbf{Sum}}(W_{R,j}) \cdot \underset{R \subset \{1..K\}, \forall j \in \hat{F}}{\mathbf{Var}}(Wb_{R,j}) \tag{2}$$

where R ($|R| = B$) is a set of the selected samples in the influential neighborhood. In Eq. 2, the first term calculates the overall contribution and the second term measures the variance of the selected B instances for every feature $\forall j \in \hat{F}$, respectively. The result of each **Sum** and **Var** function is a vector with the size of $|1 \times MaxInd(\hat{F})|$. The dot product of these two one-dimensional vectors results in a fitness value that we aim to maximize. We can see the variation of the selected samples as a weight vector for the contribution vector that encourages the selection of diverse samples. High-importance features are usually common among the samples in a neighborhood, and therefore, they have values 1 in Wb for many instnaces. If we do not weigh the overall contribution of features with their variation values, the algorithm converges early and chooses similar samples that increase the fitness value regardless of diversity consideration. Incorporating the variation ensures that features that rarely have contributions in W get attention, and thus their corresponding samples will be elected for the local explanation.

Algorithm 2 outlines our devised procedure for picking representative samples based on the genetic algorithm. Using *Initialization*, we create and evaluate the fitness of the initial population P_{It} for iteration $It = 0$ with population size $nPop$. Every individual in the population ($\forall p \in P_{It}$) is a solution vector with a length of budget B, i.e., $\forall p \in P_{It} \cdot |p| = B$. They are initialized with random values within the range $[1, K]$ to refer to the index of the potential representative samples in the neighborhood. In Eq. 2, R indicates the set of indices represented by an individual, i.e. $R = p$ where $p \in P_{It}$. Given a solution $p \in P_{It}$, the following constraint should be

Algorithm 2 Representative Sample Selection using GA

Input: Φ: fitness function, B: user's budget, K: number of neighborhood samples, I: indices of influential neighborhood, W: contribution matrix, Wb: binarized contribution matrix, $nPop$: population size, $MaxIt$: maximum number of generations, pc: cross-over percentage, pm: mutation percentage

Output: $\hat{I}$: index set of the representative instances

```
1: function GA($nPop, MaxIt, pc, pm, \Phi, B, I, W, Wb$)
2:     $It \leftarrow 0$
3:     $BestSol \leftarrow \{\}$
4:     $P_{It} \leftarrow$ Initialization($nPop, B, K, \Phi, W, Wb$)
5:     while $It < MaxIt$ do
6:         $P_{It}^{co} \leftarrow$ Selection($P_{It}, pc$)
7:         $P_{co} \leftarrow$ Cross-Over($P_{It}^{co}, \Phi, W, Wb$)
8:         $P_{It}^{mu} \leftarrow$ Selection($P_{It}, pm$)
9:         $P_{mu} \leftarrow$ Mutation($P_{It}^{mu}, \Phi, W, Wb$)
10:        $P_{merged} \leftarrow$ Merge($P_{It}, P_{co}, P_{mu}$)
11:        $P_{sorted} \leftarrow$ Sort($P_{merged}$)
12:        $P_{truncated} \leftarrow$ Truncate($P_{sorted}, nPop$)
13:        $P_{It} \leftarrow P_{truncated}$
14:        $BestSol \leftarrow P_{It}\{0\}$
15:        $P_{It+1} \leftarrow P_{It}$
16:        $It \leftarrow It + 1$
17:    $\hat{I} \leftarrow \{I_i | \forall i \in BestSol\}$
18:    return $\hat{I}$
```

satisfied: $\forall x, y \in p \cdot\ x \neq y$. It means that the solution should be a representative set with unique instances that explain different regions in the neighborhood. Using a loop block, several operations of the genetic algorithm are applied on the population until the algorithm reaches the maximum iteration, $MaxIt$. The *Selection* function chooses a portion of parents for cross-over and mutation, specified by pc and pm hyper-parameters, respectively. During the *Cross-Over* procedure, one-point cross-over at a random point is applied on every pair of the selected parents P_{It}^{co} to generate off-springs P_{co} by swapping the cross-over decision variables of the parents. The *Mutation* operator selects a random point in every selected parent P_{It}^{mu} and returns off-springs P_{mu} by changing the value of the random point with a random value $r \in [1, K]$. The original, cross-over, and mutation populations are merged together via *Merge* function results in P_{merged}. Using *Sort* operator, the population is sorted in descending order based on the fitness values and a new population is created, denoted by P_{sorted}. The final operator is *Truncate* which selects the best $nPop$ solutions ($P_{truncated}$) for the next generation (the rest of the population will be discarded). At the end of each iteration, the best solution is the first individual in the population, i.e., $P_{It}\{0\}$, which has the highest fitness value. Once the loop reaches $MaxIt$, the genetic algorithm terminates and returns the best solution $BestSol$. Representative instances $\hat{I}$ are then the neighborhood samples that are referenced by the $BestSol$.

To understand the behavior of the model in predicting an anomaly, we provide the developer with insights into the influential neighborhood by explaining the derived representative instances. These instances can be seen as prototypes for anomaly and their accumulative explanations acts as a quasi-global explanation for the neighborhood. There are several explanation techniques one can use to explain the prototypes locally. In this paper, we employ EXPLAN [7] due to its rule-based explainability style that are intuitive to comprehend. In addition to its high fidelity, precision, and stability properties, it is a computationally efficient algorithm that is suitable for generating a quasi-global explanation from multiple local explanations.

C. Global Explanation of Influential Neighborhoods

Discovering a set of samples in the training data that impact the prediction of an anomaly can provide a wide range of analysis opportunities for the developer. The set of local explanations provided in Section II-B gives detailed descriptions about the behavior of the model for individual instances. The explanations generated for the prototypes can be directly compared and correlated to the anomalous instance. Furthermore, it enables deeper and precise investigations into the neighborhood data. However, it is not adequate for specifying the general trend in the locality of the anomaly. Global explanation is suitable for discovering population level pattern such as bias and important features in a data set. Therefore, it is worth having a comprehensive understanding of the model's behavior using both complementary approaches.

Partial Dependence Plot (PDP) [9] is a well-known global explanation technique that shows the marginal effect of one or two features on the prediction of a machine learning model. A partial dependence plot demonstrates whether the relationship between the model outcome and a feature is linear, monotonic or more complex. Although partial dependence plots are intuitive and easy to implement, they have two major problems. First, the PDP does not show the distribution of the features which can be misleading, because one might over-interpret the regions with almost no data. Second, PDP assumes the feature for which partial dependence is calculated has no correlation with other features in the data. Due to this assumption, PDP creates unlikely data points which results in an biased estimation of the feature's effect.

Accumulated Local Effects (ALE) plots [10] remedies the feature distribution illustration and assumption of independence associated with PDP. Similar to PDP, it answers the following question: how does the effect of a feature on the outcome vary with the feature's value? The only distinction is in the approach of measuring the effect. PDP calculates the cumulative effects of a feature value over a marginal distribution, while ALE plot computes the same effect over a conditional distribution. In case the features are highly correlated, the marginal distribution can be much wider than conditional distribution and contain feature space regions where no likely data exists. This can be problematic because the model is not well-trained in the areas of the feature space where no training data exist, consequently leading to unrealistic prediction outcomes. The narrower conditional distribution used by ALE plots helps to mitigate this issue and make them preferable in cases where features are highly correlated. We refer the reader to [16] which provides a comparative review of PDP and ALE methods.

According to the stated advantages of ALE plot, we visualize the global effect of features in an influential neighborhood using ALE plots. Given an anomalous instance x, the indices of its impactful samples in the training data are returned, i.e., $I = N_{model}(x)$. Then, **ALE** function is applied on all features in the neighborhood for generating global explanations, i.e., $\{\mathbf{ALE}(F_i)|i = 1, .., |F|\}$, where F is the set of features in the data. The crated ALE plots can assist the feature engineering process, as features that reveal a high effect in the ALE plots can be subjected to further investigation. The plots are also helpful for finding common features with high impact among different anomalous data points. We could also identify low-importance features in the data set that unjustifiably affect the model's prediction to a great extent. Moreover, by observing the effect of the features that are amenable to bias, the potential biases in the model can be detected.

III. Experiments and Discussion

In this section, we evaluate the proposed methodology with respect to several data sets and black-box models. We evaluate the utility of the proposed framework theoretically and leave its practical applications for future work. The evaluation results are reported in three sections: A) neighborhood model analysis, B) quasi-global explanation of anomalies, and C) global explanation of influential neighborhoods.

Experimental Setup. The proposed method has been developed in Python programming language, and the experiments were run on a system with an Intel Core i7-8650U processor and 32GB of memory. We used `scikit-learn` library for implementing the machine learning and data mining algorithms [17]. Source code for replicating our experiments is available at: https://github.com/peymanrasouli/XdebugML.

In the experiments, three tabular classification data sets including *Adult*, *German credit*, and *COMPAS*[3] were used. Each data set was split into 80% *train set* and 20% *test set*. We performed experiments on the anomalous instances existing in the train set. The achieved improvement as the result of applying our technique can be validated using the test set. A Gradient Boosting classifier (**GB**) [9], a Logistic Regression classifier (**LR**) [18], and a Neural Network classifier (**NN**) [19] with the default hyper-parameters specified in the `scikit-learn` library were employed as black-box models.

A. Neighborhood Model Analysis

The aim of a neighborhood model is to identify samples in the training data that contribute mostly to the prediction of an anomalous instance. In other words, the identified samples influence the prediction of a specific anomalous data point.

[3]Data set is available at: https://www.kaggle.com/danofer/compass

TABLE II: Comparison of the neighborhood influences.

Data set	Black-box	N^c_{model}	N^f_{model}	N^p_{model}
Adult	**GB**	0.975±.1	0.548±.1	0.195±.1
	LR	0.334±.2	0.209±.1	0.077±.1
	NN	0.830±.1	0.690±.2	0.349±.2
German	**GB**	0.933±.1	0.742±.1	0.163±.1
	LR	0.990±.0	0.755±.1	0.254±.1
	NN	0.874±.1	0.773±.1	0.191±.1
COMPAS	**GB**	0.924±.1	0.625±.1	0.179±.1
	LR	0.726±.1	0.472±.2	0.106±.1
	NN	0.898±.1	0.819±.1	0.223±.1

Imagine we are capable of finding common influential samples for various groups of anomalous instances; by investigating the obtained data, we are likely to resolve the model for different groups of anomalous instances, and consequently improve the overall accuracy of the black-box model. To compare the efficacy of the devised neighborhood model (Algorithm 1), we create two other neighborhood models based on feature values and prediction probabilities of the black-box model with the same configurations. To avoid probable bias that can be caused by the nearest neighbor method, we apply feature scaling on the feature values. By convention, we denote the neighborhood models based on feature contributions, feature values, and prediction probabilities by N^c_{model}, N^f_{model}, and N^p_{model}, respectively. The number of neighborhood samples, K, for *Adult*, *German credit*, and *COMPAS* data sets is set to 2000, 200, and 500, respectively.

In this part of the experiment, we measure the influence of a retrieved neighborhood for anomalous instances appeared in the training data. The procedure is as follows: 1) for an anomalous instance $(\hat{x}, \hat{y})$ where $f(\hat{x}) \neq \hat{y}$, we achieve its influential neighborhood in the training data, i.e., $I = N_{model}(\hat{x})$; 2) the labels of the retrieved samples (indicated by I) are flipped, for example in a binary data set, if a label is 1, it is changed to 0 and vice versa; 3) we retrain the black-box model with the modified data and measure whether the label of the anomalous instance change. This procedure is conducted for 100 trials of 10 randomly selected anomalous samples in the training data and the results are reported in Table II. This metric measures the influence rate of a selected neighborhood, in other words, if the prediction of the neighborhood were changed, how much would the prediction of the anomalous instance change.

According to the reported results in the Table II, neighborhoods determined by N^c_{model} have the highest influence compared to N^f_{model} and N^p_{model} for all data sets and black-box models. We can see the localities derived from N^p_{model} impact the anomalous samples slightly. Compared to N^p_{model}, N^f_{model} has a higher performance, especially for **NN** and **GB** models. The N^c_{model} created based on feature contributions outperforms both N^p_{model} and N^f_{model}. Therefore, feature contributions provide a more precise representation of the decision boundary that allow us to identify the neighborhood of anomalous instances effectively.

Exploring the influential neighborhood of a particular

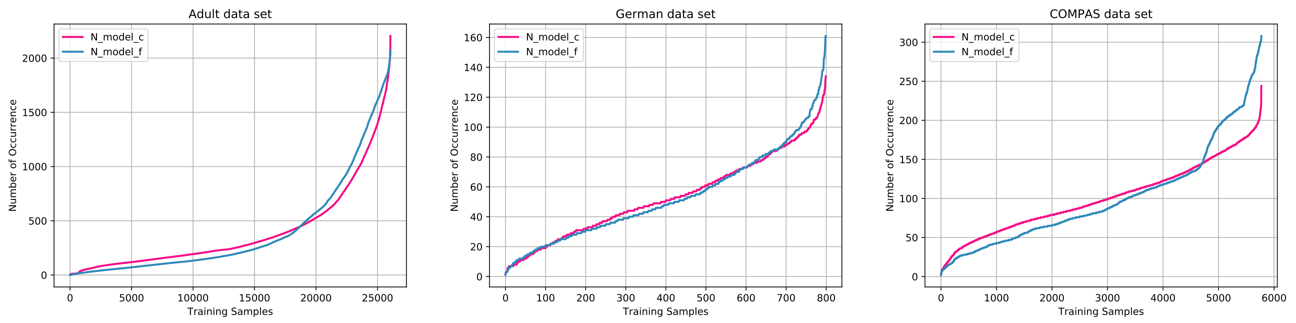

Fig. 3: Occurrence histogram of training samples in the influential neighborhoods.

anomalous instance can provide various investigation opportunities to find the causes of mispredictions. However, it becomes more useful when we find common influential samples for different categories of anomalous instances. For example, we can create clusters of anomalies based on different features, and then find samples in the training data that jointly contribute to each cluster. This operation provides valuable insights into the training data, which by engineering and refining the related data of the members of each cluster, the accuracy of the black-box model can be improved with respect to the specific groups of anomalies.

To this end, we measure the occurrence histogram of the training data for all anomalous instances. We create a histogram H with a size of $|H| = |D_{train}|$ where every bin corresponds to a training example. When a neighborhood model is invoked for an anomalous instance $\hat{x}$, i.e., $I = N_{model}(\hat{x})$, bins in H corresponding to the indices in I are incremented by the value of 1. By updating the histogram H using the locality of every anomalous instance, we achieve the common influential data points. Fig. 3 illustrates the occurrence histogram of the training data for **LR** black-box model applied on all three data sets for N^c_{model} and N^f_{model}.

Histograms plotted in Fig. 3 convey two important matters. First, the histograms of both N^c_{model} and N^f_{model} models are similar. This indicates that N^c_{model} is not biased towards selecting globally important samples for the decision boundary. Therefore, it retrieves locally relevant samples for the anomalous instances. Second, we can see there are some samples in the training data that contribute to the anomalous instances most frequently. This can effectively guide developers in finding partition of the training data that seems problematic and require further investigation/mitigation.

Mainly, there are two sources of computational complexity for the neighborhood model: feature contribution extraction and neighborhood model construction. Our framework allows using any arbitrary method for these processes. For the experiments, we had used IME contribution extraction method [12] with the explanation time complexity $O(mT_f(x))$, where m is the number of features and $T_f(x)$ is the prediction time of the model f on the instance x. We used KD tree algorithm [20] for creating the neighborhood model that has a very fast construction time of $O(mN\log(N))$ and a query time of $O(\log(N))$ where N is the number of data samples.

B. Quasi-Global Explanation of Anomalies

By observing local explanations for the anomalous instances, a domain expert can get insights about the causes of anomalies. Here, we apply our devised representative sample selection that is combined with EXPLAN method to provide local explanations for mispredicted instances in the training data. The provided set of explanations are diverse in the sense that each one explains a different spot in the influential neighborhood. The IF-THEN rules reveal locally important features for every representative instance selected for an anomalous instance and allow us to compare their explanations and understand their relations. For these experiments, we set the user's budget as $B = 10$ and the number of top features as $NF = 5$. The hyper-parameters of the genetic algorithm are set as follows: $nPop = 300$, $MaxIt = 100$, $pc = 0.8$, and $pm = 0.4$. Moreover, we run EXPLAN with its default values for hyper-parameters that are stated in the original paper.

To evaluate the efficacy of our devised sample selection procedure, we apply it to the neighborhood of 100 randomly selected anomalies in the training data, and compare the results with a random sample selection procedure. Specifically, we perform the following tasks: 1) for an anomalous instance $(\hat{x}, \hat{y})$ where $f(\hat{x}) \neq \hat{y}$, we achieve its influential neighborhood in the training data, i.e., $I = N^c_{model}(\hat{x})$; 2) Algorithm 2 is applied on the neighborhood to select B representative samples for local explanation (**GA**); 3) selected samples are explained using EXPLAN approach. We repeat the mentioned experiment for a scenario where B samples in step 2 are picked using a random selection procedure (**Random**). Results of this benchmark are evaluated with respect to the following metrics that measure the diversity of a representative set:

- Sample Diversity (**SD**): The majority of samples in the neighborhood of an anomalous instance $\hat{x}$ are either anomalous or non-anomalous with the same prediction as $\hat{x}$. This metric measures the ratio of anomalous to non-anomalous instances in a representative set. For comparison purposes, selecting a diverse set of samples is preferred (i.e., higher **SD** values).

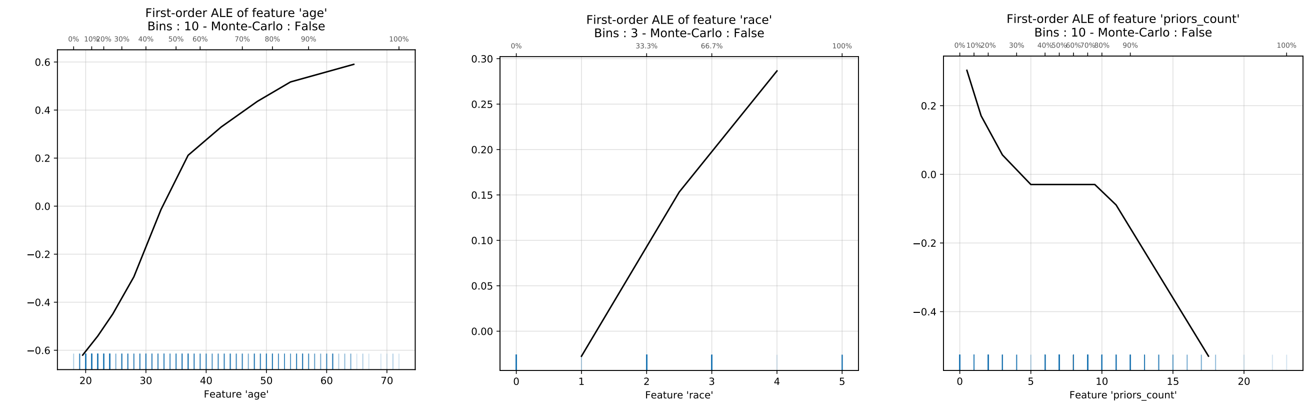

Fig. 4: Global explanation of an influential neighborhood for an anomalous instance in *COMPAS* data set with respect to features "**age**", "**race**", and "**priors_count**". The x-axis depicts possible values for a feature, and the y-axis demonstrates the overall effect of each feature value on the recidivism probability. Blue bars on the x-axis correspond to the distribution of the feature in the influential neighborhood.

TABLE III: Diversity of the quasi-global explanations.

Data set	Method	SD	FS	LD
Adult	**GA**	0.698±.2	0.578±.1	1.406±.5
	Random	0.369±.2	0.640±.1	1.272±.5
German	**GA**	0.672±.2	0.744±.1	0.435±.2
	Random	0.391±.1	0.775±.1	0.388±.1
COMPAS	**GA**	0.515±.2	0.587±.1	0.963±.4
	Random	0.327±.2	0.695±.1	0.645±.4

- Feature Similarity (**FS**): Using Jaccard coefficient we measure the similarity between the predicates (feature names) of the generated local explanations for B representative samples. We would like to have explanations that reveal different information about the influential neighborhood, therefore, having different explanations are desired (i.e., lower **FS** values).
- Length Deviation (**LD**): Decision tree models (EXPLAN's interpretable model) that capture different spots in the influential neighborhood space would have varied depth, and consequently, they would provide various length explanations. This metric measures the deviation of the length of the explanation rules. We are interested in a set of explanations with diverse lengths (i.e., higher **LD** values).

Table III reports the results of the stated benchmark for **NN** black-box model that is applied on all data sets. It exhibits that our devised genetic-based algorithm outperforms random sample selection on all defined metrics. A representative set created by **GA** selects prototype instances that lead to diverse local explanations for the various spots in the neighborhood. The generated explanations vary with respect to feature names (small **FS** values) and rule length (large **LD** values). By considering the variation among feature contributions, in the fitness function, this diversity is achieved. As indicated by **SD** metric, the ratio of anomalous to non-anomalous data points in the representative sets selected by **GA** are significantly higher than **Random** method. This diversity implies generating contrastive explanations that can guide the developer for debugging the model.

Bearing this in mind that the computational complexity of the genetic algorithm heavily depends on the fitness function, our quasi-global explanation method is very efficient as it uses a simple fitness function. The function performs three low-cost operations, i.e., summation, variance, and dot product, on pre-calculated matrices. It does not involve any time-consuming process like calling the black-box model or extracting feature contributions. As a result, computational complexity is negligibly affected by varied neighborhood sizes.

C. Global Explanation of Influential Neighborhoods

There are some questions that can not be answered using the local explanation techniques. Finding globally important features and bias existence are a few examples that require a broader understanding of the model. This demand can be fulfilled by global explanation methods that reveal general trends existing in the data and the black-box model.

We demonstrate the global insights into a retrieved influential neighborhood using ALE plots. Fig. 4 depicts three highly important features of the locality identified by N^c_{model} for an instance from *COMPAS* data set (analysis on recidivism decisions in the United States) that is mispredicted by **NN** black-box model. The ML model's task is to predict which of the bail applicants will recidivate in the next two years. It can be seen that for this particular anomalous instance, features "**age**", "**race**", and "**priors_count**" have a major effect on the overall prediction of its neighborhood. For instance, as "**age**" increases, the probability of the recidivism increases. In contrast, as "**priors_count**" increases, the probability of the recidivism decreases. The global explanation here exposes a strong evidence of bias caused by the feature "**race**". By having different values for feature "**race**", the recidivism probability increases up to 0.3. The neighborhood, hence, can be subjected to fairness analysis and refinement.

Global explanation methods sacrifice the precision for covering the whole decision boundary. To make a trade-off between precision and coverage, one can apply our neighborhood model on partitions of the data and derive a more precise explanation (rather than generating a single explanation for the entire data set). Specifically, this can be done by invoking the N^c_{model} for multiple anomalies and integrating their explanations. By doing this, we provide a global understanding of the behavior of the model by means of locally global explanations. We can repeat the mentioned procedure for non-anomalous instances, and by comparing them we are able to highlight the causes of anomalies.

IV. Conclusions

The research around developing methods for debugging and refining black-box ML models is still in its infancy. We believe such techniques can aid the developer to create a more trustworthy and reliable model. In this paper, we proposed a systematic debugging framework for developing ML models that incorporates the model's decision boundary for guiding the data engineering process. Using a neighborhood model based on feature contributions, we identify locally influential samples in the training data for a particular anomaly. By highlighting and explaining impactful data partitions for anomalies, a developer can interactively use our framework to perform various data visualization and data engineering tasks. The conducted benchmarks showed the superiority of feature contributions over feature values and prediction probabilities in finding responsible data points for anomalies. Our devised quasi-global explanation technique selects representative samples in a neighborhood to provide insights into the anomaly. Through an illustrative example, we described the utility and potential applications of such an explanation. We also equipped our framework with a robust global explanation technique to detect general trends and potential biases in the influential neighborhoods. The main application of the framework is for the development phase. Still, one can use it in the deployment phase to calibrate the prediction of new inputs through the provided explanations. We proposed several guidelines on the utility of our framework for data engineering. As future work, we validate its practical benefits on a real-world, industrial use-case.

References

[1] S. García, S. Ramírez-Gallego, J. Luengo, J. M. Benítez, and F. Herrera, "Big data preprocessing: methods and prospects," *Big Data Analytics*, vol. 1, no. 1, p. 9, 2016.

[2] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.

[3] S. Arlot, A. Celisse *et al.*, "A survey of cross-validation procedures for model selection," *Statistics surveys*, vol. 4, pp. 40–79, 2010.

[4] G. Pleiss, M. Raghavan, F. Wu, J. Kleinberg, and K. Q. Weinberger, "On fairness and calibration," in *Advances in Neural Information Processing Systems*, 2017, pp. 5680–5689.

[5] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.

[6] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[7] P. Rasouli and I. C. Yu, "Explan: Explaining black-box classifiers using adaptive neighborhood generation," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–9.

[8] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee, "From local explanations to global understanding with explainable ai for trees," *Nature machine intelligence*, vol. 2, no. 1, pp. 2522–5839, 2020.

[9] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[10] D. W. Apley and J. Zhu, "Visualizing the effects of predictor variables in black box supervised learning models," *arXiv preprint arXiv:1612.08468*, 2016.

[11] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[12] E. Strumbelj and I. Kononenko, "An efficient explanation of individual classifications using game theory," *The Journal of Machine Learning Research*, vol. 11, pp. 1–18, 2010.

[13] A. Saabas, "Treeinterpreter, https://github.com/andosa/treeinterpreter," 2020.

[14] M. Reza, B. Ghahremani, and H. Naderi, "A survey on nearest neighbor search methods," *International Journal of Computer Applications*, vol. 95, no. 25, pp. 39–52, 2014.

[15] M. Gen and L. Lin, "Genetic algorithms," *Wiley Encyclopedia of Computer Science and Engineering*, pp. 1–15, 2007.

[16] C. Molnar, "Interpretable machine learning: A guide for making black box models explainable," 2020.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[18] D. Böhning, "Multinomial logistic regression algorithm," *Annals of the institute of Statistical Mathematics*, vol. 44, no. 1, pp. 197–200, 1992.

[19] G. P. Zhang, "Neural networks for classification: a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 30, no. 4, pp. 451–462, 2000.

[20] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, p. 509–517, Sep. 1975. [Online]. Available: https://doi.org/10.1145/361002.361007