BELLA: Black-box model Explanations by Local Linear Approximations

Anonymous authors
Paper under double-blind review

Abstract

Understanding the decision-making process of black-box models has become not just a legal requirement, but also an additional way to assess their performance. However, the state of the art post-hoc explanation approaches for regression models rely on synthetic data generation, which introduces uncertainty and can hurt the reliability of the explanations. Furthermore, they tend to produce explanations that apply to only very few data points. In this paper, we present BELLA, a deterministic model-agnostic post-hoc approach for explaining the individual predictions of regression black-box models. BELLA provides explanations in the form of a linear model trained in the feature space. BELLA maximizes the size of the neighborhood to which the linear model applies so that the explanations are accurate, simple, general, and robust. BELLA can produce both factual and counterfactual explanations.

1 Introduction

Machine Learning (ML) and Artificial Intelligence (AI) models have been employed to handle tasks in various domains, including justice, healthcare, finance, self-driving cars, and many more. Consequently, legislative regulations have been proposed to protect interested parties and control the usage of these models. One example is the General Data Protection Regulation of the European Union (Goodman & Flaxman, 2017), which stipulates *the right to an explanation* in situations where an AI system has been employed in a decision-making process. The AI act (European Parliament and Council of the EU, 2024), too, has stipulated the transparency of AI models according to the level of risk they pose.

The main issue is that many ML models are *black-box models*, i.e. one cannot easily understand how they arrive at a decision. This has led to the emergence of eXplainable Artificial Intelligence (xAI), a research field that aims to make black-box models human-understandable. In this paper, we are concerned with understanding regression models, i.e., models that make a numerical prediction. We are interested in explaining a given prediction of such a model *post-hoc*, i.e., after it has been produced. This is usually done by building an interpretable surrogate-model (e.g., a decision tree) that mimics the black-box model and that can be used to understand the prediction.

Numerous approaches have been proposed to build such surrogate models, in particular SHAP (Lundberg & Lee, 2017), LIME (Ribeiro et al., 2016), and MAPLE (Plumb et al., 2018). We review them in Section 2. To evaluate the surrogate models, several criteria have been proposed: we want the surrogate model to be *accurate*, i.e., to reflect the predictions of the black-box model; we want it to be *simple*, i.e., to use few features; we want it to be *robust*, i.e., giving similar explanations to similar data points; we want it to be *general*, i.e., applicable to many data points; and we want them to be *counterfactual*, i.e., to tell us how we have to modify the data point at hand to get a different prediction. We survey these desiderata in Section 3. We find that existing approaches tend to be good on some of these criteria, but never excel on all of them. This is not surprising, as the desiderata stand in obvious conflict: A simple surrogate model, e.g., risks being not very accurate, because usually accurate predictions can be made only by the type of complex models that we wish to explain in the first place.

Our key idea (which we present in Section 4) is to train a local linear model on the neighborhood of the data point that we wish to explain. This allows us to develop an approach called BELLA (Black-box model Explanations by Local Linear Approximations). We can show through extensive experiments (in Section 5) on a dozen datasets that BELLA beats all existing approaches across nearly all desiderata.

2 Related Work

Explainable AI has received much attention in the scientific literature (Beaudouin et al., 2020; Guidotti et al., 2018; Adadi & Berrada, 2018; Murdoch et al., 2019; Burkart & Huber, 2021; Hassija et al., 2024). In this paper, we are interested in *post-hoc approaches*, i.e., those that add interpretability to a given black-box model. Some of these approaches have been developed specifically for a given type of learners (such as Gat et al. (2022) for neural models). However, we are interested in *model-agnostic* approaches, i.e., those that can interpret any black-box model. Some model-agnostic approaches compute feature importance (Chen et al., 2018; Bang et al., 2021). However, these approaches do not allow explaining unseen data points. Hence, we focus on approaches that build a *surrogate model*, i.e., a model that mimics the black-box model but that is interpretable by design (e.g., a decision tree). While *global* methods provide an interpretation of the black-box model behavior on the whole space, *local* models provide an interpretation for a single data point. In this paper, we are interested *model-agnostic post-hoc local* explanations for *regression models*, i.e., we aim to provide an explanation for a given real-valued decision by any type of model for a given data point. We are thus not interested in approaches that work for classification only (Vo et al., 2022; Mothilal et al., 2020; Bui et al., 2022; Vo et al., 2023).

One approach to deal with regression models is to adjust the methods for classification models (such as LORE (Guidotti et al., 2019)), e.g., by discretization or clustering. However, this loses information and may require domain knowledge. Therefore several approaches have been developed to natively support both classification and regression models: SHAP (Lundberg & Lee, 2017) introduces a game theory approach to compute the contribution of each feature. The explanation applies to a single data point and it is given as a linear combination of the feature contributions. In order to improve computation time, AcME (Dandolo et al., 2023) computes feature contributions based on the perturbations based on data quantiles. LIME (Ribeiro et al., 2016) generates synthetic data points by feature perturbations. This yields a weighted neighbourhood that is used to train a linear model, whose coefficients are then used as an explanation. However, both LIME and SHAP compute feature contributions in a projected, binary, space, which does not correspond to the original feature space. MAPLE (Plumb et al., 2018) addresses this problem and uses Random Forests to assign weights to the training examples. In this way, it forms a weighted neighbourhood on which the explanation applies. SHAP, LIME, and MAPLE are direct competitors to our method BELLA, and we will see in our experiments that BELLA outperforms all of them on the quality of the explanations.

DLIME (Zafar & Khan, 2019) is a deterministic variant of LIME that provides stable and consistent explanations. However, it requires extensive manual input, as the user has to provide the number of clusters for the hierarchical clustering step, the number of neighbours for the KNN step of the method, and the length of the explanation. As such, DLIME is not well suited for regression tasks and was thus applied only to classification.

Another group of approaches computes *counterfactual* explanations. One such method (White & Garcez, 2019) uses the idea of *b-counterfactuals*, i.e., the minimal change in the feature to gauge the prediction of the complex model. This method applies only to classification tasks. Another work (Dandl et al., 2020) uses Multi-Objective Optimization to compute counterfactual explanations, both for classification and regression. Another work (Redelmeier et al., 2021) uses Monte Carlo sampling for the same purpose. All of these approaches, however, can compute only counterfactual explanations, not both factual and counterfactual explanations like BELLA.

3 Preliminaries

Goal. We are given a tabular dataset $T \subset F_1 \times ... \times F_n$, where each F_i is a set of *feature values*. We are also given a function $Y : T \to \mathbb{R}$ that yields, for each $x \in T$, a *target value* $Y(x) \in \mathbb{R}$. These target values may, e.g., have been produced by a black-box model, in which case the target value is a *prediction*. Consider now one data point $x \in T$ with its target value $Y(x)$. We aim to compute an *explanation* in the following sense (Das & Rad, 2020):

Definition 1 *An explanation is additional meta information, generated by an external algorithm or by the machine learning model itself, to describe the feature importance or relevance of an input instance towards a particular output classification.*

If the target value was produced by a black-box model, we cannot be sure post-hoc that the features we identify really contributed to the computation of the target value (the model may just as well have thrown a dice, independently of any feature values). However, if several data points with these or similar feature values produce a similar prediction, we can use abductive reasoning to infer that these features may have contributed to the prediction, and that, hence, the features constitute an explanation. This is in fact common in the literature (Ribeiro et al., 2016; Lundberg & Lee, 2017; Radulovic et al., 2021; Ignatiev et al., 2019).

Quality measures. Several properties of “good” explanations have been proposed. Some of them, such as plausibility and accordance with prior beliefs, require human evaluation. Among the criteria that do not, we commonly find (Miller, 2019; Guidotti et al., 2018; Burkart & Huber, 2021; Molnar, 2018):

1. **Fidelity**: we want the value that the surrogate model explains to be close to value that the black-box model predicts.
2. **Simplicity**: we want the explanation to contain few features.
3. **Robustness**: we want similar data points to have similar explanations.

In addition, users tend to favor explanations that apply to many data points (Radulovic et al., 2021). This appears counter-intuitive, as we aim to explain only a single data point, no matter the others. And yet, it is easy to see that an explanation such as “You have a high risk of diabetes because your body mass index is 27, your A1C level is 7%, and your blood sugar level is 210mg/dL” is little satisfactory, as it allows no generalization. More helpful is to know that, *generally*, people with a body mass index larger than 25, an AIC level above 6.5%, and a blood sugar level of 200 mg/dL have a high risk of diabetes (Mayo-Clinic, 2023). We would thus like to have:

4. **Generality**: we want the number of data points to which an explanation applies to be large.

Desiderata. In addition to the above quality measures, there are also several criteria in the literature that either apply or don’t apply to a given method of explanation. One of them is (Miller, 2018; Wachter et al., 2017):

5. **Counterfactuality**: the ability to provide a set of modifications to the data point at hand that would entail a change in the decision of the black-box model.

Counterfactuality is obviously of interest to a user who wishes not just to understand the prediction, but also to actively influence it (e.g., after having been attributed a high risk of diabetes based on the results of a blood test).

Several methods for post-hoc explainability use randomization to probe the black-box model. However, this entails that the same data can lead to different explanations, which introduces uncertainty for the user (Zhang et al., 2019; Slack et al., 2020). We thus have an additional desideratum:

6. **Determinism**: the avoidance of randomization steps

Finally, some methods (Plumb et al., 2018) propose explanations that take the form of a linear equation, which allows computing the predicted value from the feature values. This is a very attractive property, as the user can toy with the explanation and apply it also to neighboring data points. We thus have a desideratum that we call

7. **Verifiability**: the possibility to compute the predicted value from the feature values

Given this plethora of quality measures and desiderata, it is not surprising that no existing method (including our own) can satisfy all of them perfectly. However, we can at least show that our method ticks all desiderata, and outperforms existing methods across nearly all quality measures.

4 BELLA

We are given a tabular dataset T, a data point $x \in T$, and a real-valued target value y, and we aim to compute an explanation for this target value. Our idea is to find a linear equation $y \approx w_1 \cdot f_1 + w_2 \cdot f_2 + \cdots + w_n \cdot f_n + w_0$, where w_i are real-valued regression coefficients and f_i are feature values of x in T. Such an equation tells the user (1) what the important features are and (2) how their can be used to compute the predicted value. To find this equation, BELLA proceeds in three steps (Algorithm 1):

1. Compute the distance of x to the other points in T.
2. Conduct a linear search to find the best neighborhood of x, according to a defined metric.
3. Train a sparse linear model on that neighborhood, and propose this model as an explanation.

Algorithm 1 BELLA

```
   Input: Dataset $T$ with labels $Y$
          Labeled data point $x \in T$
1: $d \leftarrow$ ComputeDistances($x, T$)
2: $L, N \leftarrow$ NeighborhoodSearch($x, T, d$)
3: return $L, N$
```

Step 1: Computing the distances. To compute the neighborhood of the input data point, we need a distance measure. A good starting point is to have all numerical features on the same scale so that each feature contributes to the distance measure in the same range. Therefore, we first standardize all numerical features to have a mean of 0 and a standard-deviation of 1.

To compute the distances, we employ the generalized distance function (Harikumar & Surya, 2015), which consists of three separate distance measures to account for numerical, categorical, and binary data types, as follows:

$$d(x_1, x_2) = \sum_{i=1}^{m_n} d_n(x_{1i}, x_{2i}) + \sum_{j=m_n+1}^{m_c+m_n} d_c(x_{1j}, x_{2j}) + \sum_{k=m_c+m_n+1}^{m_n+m_c+m_b} d_b(x_{1k}, x_{2k}) \tag{1}$$

Here, m_n, m_c and m_b are the number of numerical, categorical, and binary features, respectively. The distance measure for the numerical attributes d_n is the $L1$ norm $d_n(x_1, x_2) = |x_1 - x_2|$, which is preferred over $L2$, as it is more robust to outliers (Hopcroft & Kannan, 2014). For categorical features, d_c is the distance measure (Ahmad & Dey, 2007), which takes into account the distribution of values and their co-occurrence with values of other attributes. The distance between two values x and y of an attribute A_i with respect to attribute A_j is given by:

$$\delta^{ij}(x, y) = P(A_j \in \omega | A_i = x) + P(A_j \notin \omega | A_i = y) - 1$$

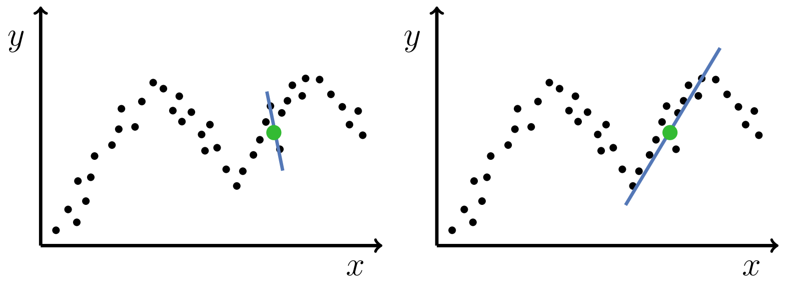


Figure 1: Left: an explanation for a data point x that is too specific, applying only to a very small neighborhood. Right: An explanation that applies to a larger neighborhood, which is what we aim at.

Here, $P(A_j \in \omega | A_i = x)$ is the conditional probability that attribute A_j will take a value from the set ω given that the attribute A_i takes the value x. ω is a subset of all possible values of attribute A_j that maximizes the sum of the probabilities. Since both probabilities can take values from $[0, 1]$, we subtract 1 in order to arrive at $\delta^{ij}(x, y) \in [0, 1]$. Lastly, for binary features, we use the Hamming distance: $d_h(a, b) = 1$ if $a = b$, and zero otherwise. In Line 1 of Algorithm 1, the function *ComputeDistances* returns the distances by Equation 1.

Step 2: Neighborhood Search. After computing the distances, we proceed with the exploration of the neighborhood of the input data point x. The goal is to find a set of points, closest to x according to the distance measure, that will serve as a training set for a local surrogate model. Several common techniques could be considered to that end, including kNN, K-Means, and other distance-based clustering methods. In our case, however, we aim to find a neighborhood such that a linear regression model trained on that neighborhood represents an accurate local approximation of the black-box model. Hence, the quality of the neighborhood is proportional to the quality of the performance of the linear model fitted on it. Common drawbacks of regression evaluation metrics metrics are missing interpretability, sensitivity to outliers and near-zero values, divisions by zero, missing bounds, and missing symmetry. We find that the *Berry-Mielke universal R value* $\Re$ (Berry & Mielke Jr, 1988) avoids most of these pitfalls. $\Re$ represents the measure of agreement between raters and it is a generalization of Cohen's kappa (Cohen, 1960). $\Re$ measures how much better the model is compared to a naive one (e.g., to a random predictor). $\Re$ takes values from the range $[0, 1]$, and it can be interpreted easily: If $\Re$ is equal to 0, the model performance is equal to the one of the random model and if it is 1, then the model has perfect performance. $\Re$ is defined as $\Re = 1 - \frac{\delta}{\mu}$, where δ and μ are defined as:

$$\delta = \frac{1}{n} \sum_{i=1}^{n} \Delta(\hat{y}_i, y_i), \quad \mu = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \Delta(\hat{y}_j, y_i). \tag{2}$$

Here, n is the number of samples, y_i is the actual target value, $\hat{y}_i$ is the predicted value, and $\Delta(\cdot)$ represents the distance function between the true and the predicted value. The original work by (Berry & Mielke Jr, 1988) uses the Euclidean distance, but later works (Janson & Olsson, 2001; 2004) propose to use the squared Euclidean distance instead, because this distance is equivalent to the variance of the variable, which further improves the interpretability of $\Re$. We follow this argumentation, and use $\Delta(x, y) = (x - y)^2$. This definition implies that Δ is in fact equal to the Mean Squared Error (MSE). Thus, by optimizing $\Re$, we are actually optimizing the accuracy of the local model.

However, to avoid explanations that are too specific, i.e., explanations that apply to very small neighborhoods, as in Figure 1 (left), we wish to optimize not just the *accuracy*, but also the *generality* of the surrogate model. Therefore we include the size of the neighborhood in the optimization function, to aim for explanations that are at the same time accurate and general (Figure 1 (right)). One way to do this is to maximize the lower bound of the confidence interval of $\Re$. The lower and upper bounds for the confidence interval of $\Re$ are given by (Berry & Mielke Jr, 1988):

$$CI_{\Re} = \overline{\Re} \pm MOE_{\Re} = 1 - \frac{\overline{\delta} \mp MOE_{\delta}}{\mu} \tag{3}$$

Here, *MOE* stands for the Margin of Error. From Equation 3, it follows that computing the lower bound of $\Re$ is analogous to computing the upper bound of δ. Therefore, we can compute the margin of error for δ as $MOE_\delta = t\frac{\sigma}{\sqrt{n}}$, where σ is the standard deviation of the sample, n is the sample size and t represents the critical value from the t-distribution. We use the t-distribution because it is adapted for small sample sizes, which is what we encounter when we grow the neighborhood. The distribution converges to the normal distribution as the sample size increases.

Due to the non-monotonic nature of the $\Re$ value, we have to explore the whole space to maximize its lower bound. We employ a linear search algorithm (Algorithm 2) to this end.

Algorithm 2 Neighborhood Search

```
    Input: Labeled data point $x \in T$
           Dataset $T$ with labels $Y$
           Distances $d : T \rightarrow \mathbb{R}$ of the data points to $x$
 1: Sort $T$ by ascending $d$
 2: $n \leftarrow$ number of features in $T$
 3: $max\Re_{lb} \leftarrow 0, bestN \leftarrow 0, bestL \leftarrow \emptyset$
 4: for $i = min(2n, |T|)$ to $|T|$ do
 5:     $L \leftarrow TrainLocalSurrogateModel(T[0:i])$
 6:     if $\Re_{lb}(L) > max\Re_{lb}$ then
 7:         $max\Re_{lb} \leftarrow \Re_{lb}(L)$, $bestN \leftarrow i$, $bestL \leftarrow L$
 8:     end if
 9: end for
10: return $bestL$, $T[0:bestN]$
```

The algorithm receives as input a labeled data point x that is to be explained, a labeled training set T, and a vector of distances between x and each point in the training set T. We sort the training set by increasing distance to x, train a linear model on the first i data points for increasing i, and return the set of neighbors for which the lower bound of $\Re$ is maximal. As the neighbourhood is very small in the beginning, the training easily lead to overfitting. Therefore, we consider at least $2n$ data points for our neighborhood, where n is the number of features. This ensures that the estimation of regression coefficients exhibits less than 10% relative bias (Austin & Steyerberg, 2015).

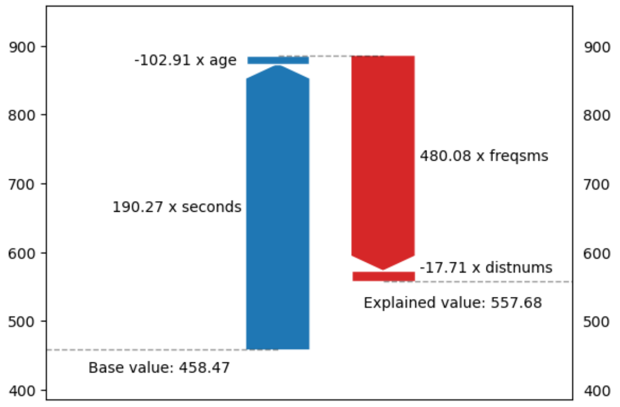
Algorithm 3 Train Local Surrogate Model

```
   Input: Neighborhood of data points $N$
1: $F \leftarrow$ the set of all features in $N$
2: $F' \leftarrow \{f | f \in F \wedge VIF(f) < 10.0\}$
3: $Features_{Lasso} \leftarrow Lasso(cv = 5, features = F')$
4: return $OLS(Features_{Lasso})$
```

Step 3: Building a local surrogate model. We build our local surrogate model on the neighborhood we have found. To obtain a model with few parameters (i.e., a *simple* model), we use regularization. In terms of feature selection, $L1$ regularization (e.g. Lasso (Hastie et al., 2009)) is able to select a nearly perfect subset of variables in a wide range of situations. The only condition for this to work is that there are no highly collinear variables (Candès & Plan, 2009), which can significantly reduce the precision of estimated regression coefficients. To remove highly collinear features, we compute the *variance inflation factor* (VIF), and, following a rule of thumb (Stine, 1995), adopt 10 as the cut-off value for the VIF.

After removing highly collinear features, the next step is to train a linear model with Lasso regularization. Lasso regularization adds a penalty term in the form of the sum of absolute values of regression coefficients. The objective function is $\min_{\beta \in \mathbb{R}^p}(||y - \beta X||_2^2 + \lambda||\beta||_1)$, where λ is the shrinkage parameter. This provides a sparse model, by forcing some coefficients to be zero. Removing some features ensures a better generalization, and results in simpler, and thus more comprehensible explanations. On the other hand, coefficients obtained by minimizing the Lasso objective function are biased towards zero. Therefore, Lasso is preferred for model

Figure 2: Explanation example.



The value predicted by the model is **551** and the explained value is **557**.
This explanation applies to **476** other instances.

selection rather than for prediction. The common strategy is to train an Ordinary Least Squares (OLS) linear model on the subset of variables selected by Lasso. This corresponds to a special variation of the relaxed Lasso (Meinshausen, 2007), with $\phi = 0$.

To determine the value of the shrinkage coefficient λ, we use 5-fold cross-validation (CV). To preserve the deterministic nature we perform CV on adjacent slices of the dataset, without random shuffles. CV selects the best model in terms of a prediction error. Since the goal of this step is model selection, we want to avoid choosing λ too small, and hence we apply the common one-standard-error rule. According to this rule, the most parsimonious model is the one whose error is no more than one standard error above the error of the best model (Hastie et al., 2009).

Once we have obtained the most parsimonious model, i.e., the best set of features, we train the final local surrogate model as an OLS model using the features selected by Lasso. This procedure is described in Algorithm 3, and it returns a local linear model.

Providing an explanation. As the final result, BELLA outputs the OLS model computed by Algorithm 3, together with the size of the neighborhood. As an example, consider the Iranian Churn dataset (Jafari-Marandi et al., 2020). It contains the (anonymized) customers of a telecommunication company, with their age, subscription length, the satisfaction with the service, etc. The goal is to predict the commercial value of the customer to the company (in dollars).

Let us now consider a given customer, for which a black-box model predicted a commercial value of \$551. The explanation that BELLA can provide for this prediction is shown in Figure 2. All numerical features have been standardized to have mean value equal to 0 and standard deviation equal to 1. (Thus, a customer has a "negative age" if they are younger than the average customer.) In the explanation, the base value is the output of the model when all inputs are set to zero (i.e., to their mean value). Each bar shows the total contribution of each feature to the predicted value: The more the customer phones (variable *seconds*), the more revenue the company generates. The age (which is below average for this particular customer), likewise, has a small positive impact. The number of SMS, in contrast, (variable *freqSMS*) impacts the revenue negatively. Finally, the number of distinct phone numbers called (variable *distnum*) has a small negative impact. These sizes of the bars are easy to interpret: The size of each bar is equal to the value of the feature multiplied by the weight computed by our method. Their sum is then directly equivalent to the explained value:

$$y \approx 458.47 + 190.27 \times \mathit{seconds} - 102.91 \times \mathit{age} + 480.08 \times \mathit{freqSMS} - 17.71 \times \mathit{distNums}$$

This computation applies to all data points in the neighborhood of the input data point (to the current instance and 476 others in our example). We thus see that BELLA's explanations are *verifiable* (because they take the form of a linear equation), *deterministic* (because BELLA does not use any randomized steps), *simple* (because we applied regularization), *general* (because we maximized the neighborhood), and *accurate* (because we optimized the linear model on the local neighborhood). In addition, BELLA does not probe the black-box model. This means that, unlike many of its competitors, BELLA can explain not just the decisions of a black-box model, but any numerical variable in a tabular dataset – even if that variable was not generated by a model at all but merely observed in reality (such as, e.g., housing prices). Let us now turn to the missing desideratum, *counterfactuality*.

Counterfactual explanations provide information about a (minimal) change needed to alter the prediction of the black-box model. In a classification scenario, the goal is to make the model predict a different class. In a regression scenario, the goal is not to make the model predict *any other value*, but the value that the user would like to see. In our example of the Iranian churn dataset, an analyst may ask why the model predicted \$551 instead of, say, \$1000. A counterfactual explanation should suggest a set of changes that should be applied to reach this reference value. To provide such an explanation with BELLA, we select candidates, i.e. data points whose target value is in an ϵ-vicinity to the reference value (with $\epsilon = 5\%$). There can be multiple candidates.

To find the best one, we optimize two criteria: the distance between the given data point and the candidate, and the amount of change needed. The first criterion will favor candidates that are in the vicinity of the given data point. The second criterion controls the amount of change applied. To alter the outcome, one can usually either apply a small change to several features, or a big change to few features. The second approach is risky: without human intervention, we can end up with a set of features that are difficult or impossible to change (e.g., the age of a customer). Therefore, we rather aim to minimize the average amount of change and suggest smaller adjustments to multiple features (such as frequency of use, or the number of SMS messages). This yields the following objective function:

$$\min_{x_i \in S}(d(x, x_i) + \frac{d(x, x')}{|\Delta|}) \tag{4}$$

Here, x_i is a counterfactual candidate data point, d is a distance measure defined in Equation 1, x' represents the modified data point x according to the counterfactual explanation and $|\Delta|$ is the number of features that have been modified. The modified data point, x', represents the counterfactual explanation.

The overall process is described in Algorithm 4. The algorithm takes as input the labeled dataset T, a labeled datapoint $x \in T$, a reference value $y_{ref} \in \mathbb{R}$, and a permitted deviation ϵ from the reference value. We first choose the set of counterfactual candidates X_c that have the target value in the ϵ neighbourhood of the reference value y_{ref}. For each x_i among these candidates, we compute the explanation using BELLA. This explanation gives us a set of features, and the proposed modification of x is to set all these features of x to the values given by x_i. Among these proposed modifications, we choose the one that minimizes the objective function in Equation 4.

Algorithm 4 Computing a counterfactual explanation

```
    Input: Dataset $T$ of data points $x_i$ with labels $y_i$
           Labeled data point $x \in T$
           A reference value $y_{ref} \in \mathbb{R}$
           Deviation from reference value $\epsilon = 0.05$
1: $X_c \leftarrow \{x_i \in T : y_i \in [y_{ref} - \epsilon, y_{ref} + \epsilon]\}$
2: for $x_i \in X_c$ do
3:     $L_i, N_i \leftarrow \text{BELLA}(T, x_i)$                    ▷ Algorithm 1
4:     $x'_i \leftarrow x_i$ modified according to $L_i$
5: end for
6: $x_{ref} \leftarrow argmin_{x'_i}(d(x, x_i) + \frac{d(x, x'_i)}{|\Delta|})$
7: return $x_{ref}$
```

Table 1: Regression Datasets

Dataset	Features	Numerical	Categorical	Instances
Auto MPG	7	6	1	392
Bike	12	9	3	8760
Concrete	8	8	0	1030
Servo	4	0	4	167
Electrical	12	12	0	10000
Superconductivity	81	81	0	21262
White Wine Quality	11	11	0	4898
Real Estate Valuation	5	5	0	414
Wind	14	14	0	6574
CPU activity	12	12	0	8192
Echocardiogram	9	6	3	17496
Iranian Churn	11	8	3	3150

5 Experiments

Datasets. We performed experiments on datasets from two standard repositories (Dua & Graff, 2017; Romano et al., 2021), shown in Table 1. Among them is also a high-dimensional dataset, Superconductivity, with 81 features. All categorical features have been one-hot encoded and all numerical features have been standardized. To show that BELLA works with different families of models, we trained a random forest (with 1000 trees), and a neural network (with one hidden layer with 500 nodes) as black-box models. For space reasons, we show only results for the neural network while the results for the random forest are in the supplementary material.

BELLA. Our method is implemented in Python. For the black-box models, we use the implementations of `scikit-learn` (Pedregosa et al., 2011). All experiments are run on a Fedora Linux (release 38) computer with an Intel(R) Xeon(R) v4 @ 2.20GHz CPU, a memory of 64 GB, and Python 3.9. All code and the data for BELLA and the experiments is available on Github (URL masked for anonymity).

Competitors. We compare BELLA to LIME (Ribeiro et al., 2016), SHAP (Lundberg & Lee, 2017) and MAPLE (Plumb et al., 2018). We use the implementations by the authors[1][2][3]. We do not compare to methods that are designed for classification tasks, or that can provide only counter-factual explanations and not factual ones (see again Section 2).

5.1 Experimental results

We compare BELLA's performance against the competitors on the quality measures from Section 3. All tables show the average performance on the test set of each method with confidence intervals at $\alpha = 95\%$.

Fidelity is measured by the Root Mean Squared Error (RMSE) of the local surrogate models wrt. the predictions of the black-box models (Table 2, with a min-max normalized average). SHAP always has an error of 0. This is because it provides exact explanations that apply only to a single data point. Among the methods that apply to a neighborhood of points, MAPLE is generally the best, followed closely by BELLA. LIME comes last. Among all these methods, BELLA is the only one that can provide both factual and counterfactual explanations. To evaluate the quality of BELLA's counterfactual explanations, we measure their fidelity wrt. a reference value. For each data point x with its target value y, we set the reference values to $y_{ref} = y \pm 0.3 \times |y_{max} - y_{min}|$. We see that the counterfactual explanations of BELLA are often of similar fidelity as its factual explanations. Even in the cases where the error of counterfactual explanations

[1]https://github.com/marcotcr/lime
[2]https://github.com/slundberg/shap
[3]https://github.com/GDPlumb/MAPLE/tree/master

Table 2: Fidelity comparison (RMSE – smaller is better)

	Factual				Counterf.
Dataset	LIME	MAPLE	BELLA	SHAP	BELLA
Auto MPG	2.16±0.487	1.04±0.239	**0.75**±0.245	**0.00**	3.74±1.100
Bike	392.00±23.70	**67.60**±5.110	164.00±11.00	**0.00**	474±20.72
Concrete	11.70±1.540	**2.30**±0.382	3.71±0.554	**0.00**	**1.23**±0.410
Servo	0.87±0.289	**0.26**±0.092	0.88±0.197	**0.00**	0.89±0.203
Electrical	0.03±0.002	**0.01**±0.001	0.02±0.001	**0.00**	**0.03**±0.001
Supercond.	25.20±1.271	**2.56**±0.357	6.84±0.978	**0.00**	58.00±1.860
White Wine	0.35±0.041	**0.19**±0.026	**0.19**±0.023	**0.00**	0.86±0.058
Real Estate	5.22±1.350	1.83±0.641	**0.84**±0.511	**0.00**	8.76±2.231
Wind	3.85±0.539	**1.28**±0.233	1.45±0.231	**0.00**	4.62±0.204
CPU Activity	18.70±1.700	**0.75**±0.058	0.92±0.114	**0.00**	**2.76**±0.370
Echocard.	9.56±0.250	**2.09**±0.087	2.92±0.119	**0.00**	13.1±0.422
Iranian Churn	147.00±20.70	**2.82**±0.468	14.30±2.530	**0.00**	**28.00**±5.970
Norm. avg.	0.11±0.008	0.03±0.003	0.04±0.006	0.00	0.12±0.047

Table 3: Generality comparison (% - larger is better)

Dataset	LIME	SHAP	MAPLE	BELLA
Auto MPG	10.08±2.520	0.00	**38.33**±3.100	37.27±8.710
Bike	4.14±0.458	0.00	5.42±0.088	**32.11**±2.160
Concrete	1.10±0.400	0.00	**31.75**±1.000	21.55±4.142
Servo	16.82±4.130	0.00	74.35±3.730	**76.31**±9.410
Electrical	0.02±0.011	0.00	20.67±0.321	**21.61**±3.180
Supercond.	0.01±0.709	0.00	15.98±0.498	**30.25**±3.480
White Wine	0.83±0.243	0.00	**17.89**±0.369	15.34±1.950
Real Estate	3.75±1.710	0.00	**46.34**±3.990	16.21±6.410
Wind	0.24±0.044	0.00	12.69±0.186	**94.00**±1.520
CPU Activity	0.71±0.250	0.00	9.30±0.227	**17.50**±1.610
Echocard.	0.00±0.001	0.00	5.13±0.060	**83.76**±1.180
Iranian Churn	0.65±0.202	0.00	**12.10**±0.417	6.41±2.140
Average	3.21±1.190	0.00	24.16±1.166	**37.95**±3.813

is an order of magnitude larger, it is still lower or comparable to the error of the factual-only explanations provided by LIME (shown in bold).

Generality is measured by the number of data points to which the explanation applies (as a percentage of all data points in the training set). For BELLA, we simply return the size of the neighborhood. For MAPLE we return the number of data points that have weights larger than 0. For LIME, an explanation comes with the range of values for each feature. We count the number of data points that fall into this range. The results are shown in Table 3. For SHAP, the size of the neighborhood is always 0. This is because SHAP provides feature contributions that are specific for the given data point, and there is no way to apply these explanations to other data points. LIME's explanations are more general, and MAPLE's explanations even more. Still, they are vastly less general than the explanations of BELLA.

Simplicity is most commonly measured by the number of features that an explanation contains (Table 4). LIME has the same size of explanations as BELLA. This is because LIME takes this parameter as input and we set it to the size of the explanation provided by BELLA. SHAP and MAPLE constantly provide longer explanations than BELLA. MAPLE has higher complexity than SHAP, even though it comes with lower accuracy.

Table 4: Simplicity comparison (smaller values are better). LIME requires the explanation size as input, and we give it the size of the explanation computed by BELLA.

Dataset	SHAP	MAPLE	BELLA/LIME
Auto MPG	9.00$\pm$0.000	6.75$\pm$0.208	**4.90**$\pm$0.369
Bike	11.63$\pm$0.038	12.44$\pm$0.080	**7.30**$\pm$0.115
Concrete	8.00$\pm$0.000	7.00$\pm$0.000	**5.42**$\pm$0.297
Servo	13.06$\pm$0.218	15.71$\pm$0.155	**5.06**$\pm$0.645
Electrical	12.00$\pm$0.000	12.00$\pm$0.000	**8.63**$\pm$0.087
Supercond.	77.97$\pm$0.246	79.99$\pm$0.008	**13.10**$\pm$0.555
White Wine	11.00$\pm$0.000	10.00$\pm$0.000	**7.54**$\pm$0.198
Real Estate	5.00$\pm$0.000	**4.00**$\pm$0.000	4.05$\pm$0.327
Wind	13.65$\pm$0.132	13.00$\pm$0.000	**9.47**$\pm$0.102
CPU Activity	12.00$\pm$0.000	12.00$\pm$0.000	**10.80**$\pm$0.204
Echocardiogram	9.00$\pm$0.000	8.47$\pm$0.026	**8.23**$\pm$0.037
Iranian Churn	9.15$\pm$0.054	9.53$\pm$0.059	**5.65**$\pm$0.166
Norm. Avg.	0.92$\pm$0.002	0.91$\pm$0.004	**0.59**$\pm$0.022

Table 5: Robustness comparison (0 to 1 – larger is better)

Dataset	LIME	SHAP	MAPLE	BELLA
Auto MPG	0.78$\pm$0.023	0.68$\pm$0.083	**0.84**$\pm$0.037	0.80$\pm$0.038
Bike	**0.79**$\pm$0.026	0.70$\pm$0.037	0.66$\pm$0.033	0.76$\pm$0.050
Concrete	**0.78**$\pm$0.047	0.72$\pm$0.028	0.68$\pm$0.041	0.65$\pm$0.081
Servo	**0.77**$\pm$0.017	0.59$\pm$0.021	0.56$\pm$0.097	0.64$\pm$0.029
Electrical	0.63$\pm$0.015	0.59$\pm$0.022	**0.76**$\pm$0.022	**0.76**$\pm$0.041
Supercond.	0.89$\pm$0.014	0.87$\pm$0.031	0.58$\pm$0.076	**0.93**$\pm$0.059
White Wine	**0.67**$\pm$0.035	0.56$\pm$0.051	0.66$\pm$0.030	0.65$\pm$0.064
Real Estate	0.70$\pm$0.057	0.74$\pm$0.041	**0.77**$\pm$0.058	0.65$\pm$0.085
Wind	0.64$\pm$0.037	0.62$\pm$0.035	0.67$\pm$0.023	**0.99**$\pm$0.109
CPU Activity	0.46$\pm$0.034	0.73$\pm$0.035	0.76$\pm$0.031	**0.83**$\pm$0.039
Echocardiogram	0.75$\pm$0.039	0.66$\pm$0.034	0.55$\pm$0.039	**0.96**$\pm$0.017
Iranian Churn	0.62$\pm$0.035	0.79$\pm$0.045	**0.84**$\pm$0.039	0.76$\pm$0.049
Average	0.71$\pm$0.032	0.69$\pm$0.039	0.68$\pm$0.044	**0.78**$\pm$0.055

Robustness judges how similar the explanations for close data points are. We measure robustness as:

$$robustness = 1 - \frac{1}{n}\sum_{i=1}^{n}\frac{|\beta_{1i} - \beta_{2i}|}{|\beta_{1i}| + |\beta_{2i}|}. \quad (5)$$

Here, n is the number of features, and β_{1i} and β_{2i} are the weights of feature i in the first and second explanation, respectively. Robustness is in the range of $[0, 1]$, with 1 indicating that two explanations are identical. We compute explanations for each data point in the test set, and compute robustness wrt. the 10 closest data points (Table 5). LIME samples 5000 data points to create a synthetic neighborhood. Thus, LIME can perform slightly better than our approach on datasets that have fewer observations. Still, in the majority of cases, and on average, BELLA outperforms LIME. BELLA also outperforms SHAP by a wide margin. This is because SHAP's explanations are tailored for a single data point. BELLA also outperforms MAPLE. This is because the crisp neighbourhood of BELLA provides much more robust explanations than MAPLE's weighted neighbourhood.

The results when the black box model is a random forest are shown in the appendix, and they do not differ much. From Tables 2, 3, 4, and 5, we can see that at the same level of simplicity, BELLA provides more general, more robust, and more accurate explanations than LIME. BELLA provides less accurate explanations than SHAP and MAPLE, but at the same time, BELLA's explanations are more general, more robust, and vastly simpler.

Counterfactuality is a desideratum that only BELLA, and neither LIME nor SHAP nor MAPLE fulfills. To evaluate the quality of BELLA's counterfactual explanations, we measure the fidelity of an explanation wrt. a reference value. For each data point x with its target value y, we set the reference values to $y_{ref} = y \pm 0.3 \times |y_{max} - y_{min}|$. The last column of Table 2 shows the RMSE of the counterfactual explanations. We see that the counterfactual explanations of BELLA are often of similar fidelity as its factual explanations. Even in the cases where the error of counterfactual explanations is an order of magnitude larger, it is still lower or comparable to the error of the factual-only explanations provided by LIME.

Other desiderata outlined in Section 3 were determinism, and verifiability (the possibility to compute the explained value from the feature values). SHAP offers none of these. Neither does LIME. While both SHAP and LIME compute linear models with feature weights, these models are not verifiable in our sense: There is no way that the user can insert the feature values of a neighboring point into these models and obtain an explained value. This is because the linear models do not operate in the original input feature space. Only MAPLE offers this verifiability. However, it relies on randomization and provides no counterfactuality. BELLA is thus the only approach that delivers deterministic, verifiable, and both factual and counterfactual explanations.

Verification on an interpretable model. To confirm that the explanations provided by BELLA represent what the black-box model has learned, we evaluate them with regard to an already interpretable model. Instead of a black-box model, we train an Ordinary Least Square linear regression model and consider the 5 most important features. We then compute the explanations for each data point in the test set with our method. BELLA was able to recover on average 85.12% of the original top-5 features across all datasets. This shows that our method provides explanations that generally agree with prior beliefs, as encoded in an interpretable model.

6 Conclusion

We have presented BELLA, an approach to provide post-hoc local explanations for any regression black-box model, or indeed any static tabular dataset with a numeric variable to be explained. BELLA is deterministic, and can provide both factual and counterfactual explanations. BELLA's objective function ensures accurate, general, robust, and simple explanations. Detailed experiments show that BELLA outperforms state-of-the-art approaches on these desiderata, often by a wide margin.

Future work could investigate how human intervention could lead to more plausible explanations. For example, our counterfactual explanations could be improved if users specified which features can be modified. We hope that our work can open the door to research along this line and others, and ultimately make machine learning models more interpretable.

References

Amina Adadi and Mohammed Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *Access*, 6, 2018.

Amir Ahmad and Lipika Dey. A k-mean clustering algorithm for mixed numeric and categorical data. *Data & Knowledge Engineering*, 63(2):503–527, 2007.

Peter C Austin and Ewout W Steyerberg. The number of subjects per variable required in linear regression analyses. *Journal of clinical epidemiology*, 68(6):627–636, 2015.

Seojin Bang, Pengtao Xie, Heewook Lee, Wei Wu, and Eric Xing. Explaining a black-box by using a deep variational information bottleneck approach. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11396–11404, 2021.

Valérie Beaudouin, Isabelle Bloch, David Bounie, Stéphan Clémençon, Florence d'Alché Buc, James Eagan, Winston Maxwell, Pavlo Mozharovskyi, and Jayneel Parekh. Flexible and context-specific ai explainability: a multidisciplinary approach. *SSRN*, 3559477, 2020.

Kenneth J Berry and Paul W Mielke Jr. A generalization of cohen's kappa agreement measure to interval measurement and multiple raters. *Educational and Psychological Measurement*, 48(4):921–933, 1988.

Ngoc Bui, Duy Nguyen, and Viet Anh Nguyen. Counterfactual plans under distributional ambiguity. *arXiv preprint arXiv:2201.12487*, 2022.

Nadia Burkart and Marco F Huber. A survey on the explainability of supervised machine learning. *JAIR*, 70:245–317, 2021.

Emmanuel J Candès and Yaniv Plan. Near-ideal model selection by $l1$ minimization. *The Annals of Statistics*, 37(5A):2145–2177, 2009.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *International conference on machine learning*, pp. 883–892. PMLR, 2018.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20 (1):37–46, 1960.

Susanne Dandl, Christoph Molnar, Martin Binder, and Bernd Bischl. Multi-objective counterfactual explanations. In *International Conference on Parallel Problem Solving from Nature*, pp. 448–469. Springer, 2020.

David Dandolo, Chiara Masiero, Mattia Carletti, Davide Dalle Pezze, and Gian Antonio Susto. Acme—accelerated model-agnostic explanations: Fast whitening of the machine-learning black box. *Expert Systems with Applications*, 214:119115, 2023.

Arun Das and Paul Rad. Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint*, arXiv:2006.11371, 2020.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

European Parliament and Council of the EU. Regulation (eu) 2024/1689 of the european parliament and of the council of 13 june 2024 laying down harmonised rules on artificial intelligence and amending regulations (ec) no 300/2008, (eu) no 167/2013, (eu) no 168/2013, (eu) 2018/858, (eu) 2018/1139 and (eu) 2019/2144 and directives 2014/90/eu, (eu) 2016/797 and (eu) 2020/1828 (artificial intelligence act), 2024. URL `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:L_202401689`.

Itai Gat, Nitay Calderon, Roi Reichart, and Tamir Hazan. A functional information perspective on model interpretation. In *International Conference on Machine Learning*, pp. 7266–7278. PMLR, 2022.

Bryce Goodman and Seth Flaxman. European union regulations on algorithmic decision-making and a "right to explanation". *AI magazine*, 38(3), 2017.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys*, 51(5), 2018.

Riccardo Guidotti, Anna Monreale, Fosca Giannotti, Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. Factual and counterfactual explanations for black box decision making. *IEEE Intelligent Systems*, 34(6): 14–23, 2019.

Sandhya Harikumar and PV Surya. K-medoid clustering for heterogeneous datasets. *Procedia Computer Science*, 70:226–237, 2015.

Vikas Hassija, Vinay Chamola, Atmesh Mahapatra, Abhinandan Singal, Divyansh Goel, Kaizhu Huang, Simone Scardapane, Indro Spinelli, Mufti Mahmud, and Amir Hussain. Interpreting black-box models: a review on explainable artificial intelligence. *Cognitive Computation*, 16(1):45–74, 2024.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

John Hopcroft and Ravi Kannan. *Foundations of data science.* Cambridge University Press;, 2014.

Alexey Ignatiev, Nina Narodytska, and Joao Marques-Silva. Abduction-based explanations for machine learning models. In *AAAI*, volume 33-01, pp. 1511–1519, 2019.

Ruholla Jafari-Marandi, Joshua Denton, Adnan Idris, Brian K Smith, and Abbas Keramati. Optimum profit-driven churn decision making: innovative artificial neural networks in telecom industry. *Neural Computing and Applications*, 32:14929–14962, 2020.

Harald Janson and Ulf Olsson. A measure of agreement for interval or nominal multivariate observations. *Educational and Psychological Measurement*, 61(2):277–289, 2001.

Harald Janson and Ulf Olsson. A measure of agreement for interval or nominal multivariate observations by different sets of judges. *Educational and Psychological Measurement*, 64(1):62–70, 2004.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NEURIPS*, 2017.

Mayo-Clinic. Diabetes, 2023. URL `https://www.mayoclinic.org/diseases-conditions/diabetes/diagnosis-treatment/drc-20371451`.

Nicolai Meinshausen. Relaxed lasso. *Computational Statistics & Data Analysis*, 52(1):374–393, 2007.

Tim Miller. Contrastive explanation: A structural-model approach. *arXiv preprint*, arXiv:1811.03163, 2018.

Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.

Christoph Molnar. A guide for making black box models explainable. *URL: https://christophm. github. io/interpretable-ml-book*, 2:3, 2018.

Ramaravind K Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pp. 607–617, 2020.

W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Interpretable machine learning: definitions, methods, and applications. *arXiv preprint*, arXiv:1901.04592, 2019.

F. Pedregosa, G. Varoquaux, Gramfort, et al. Scikit-learn: Machine learning in Python. *JMLR*, 12, 2011.

Gregory Plumb, Denali Molitor, and Ameet S Talwalkar. Model agnostic supervised local explanations. *NeurIPS*, 31, 2018.

Nedeljko Radulovic, Albert Bifet, and Fabian Suchanek. Confident interpretations of black box classifiers. In *(IJCNN)*, pp. 1–8. IEEE, 2021.

Annabelle Redelmeier, Martin Jullum, Kjersti Aas, and Anders Løland. Mcce: Monte carlo sampling of realistic counterfactual explanations. *arXiv preprint arXiv:2111.09790*, 2021.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you? – explaining the predictions of any classifier. In *SIGKDD*, 2016.

Joseph D Romano, Trang T Le, William La Cava, John T Gregg, Daniel J Goldberg, Praneel Chakraborty, Natasha L Ray, Daniel Himmelstein, Weixuan Fu, and Jason H Moore. Pmlb v1.0: an open source dataset collection for benchmarking machine learning methods. *arXiv preprint*, arXiv:2012.00058v2, 2021.

Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *AAAI/ACM Conference on AI, Ethics, and Society*, pp. 180–186, 2020.

Robert A Stine. Graphical interpretation of variance inflation factors. *The American Statistician*, 49(1): 53–56, 1995.

Vy Vo, Van Nguyen, Trung Le, Quan Hung Tran, Gholamreza Haffari, Seyit Camtepe, and Dinh Phung. An additive instance-wise approach to multi-class model interpretation. *arXiv preprint arXiv:2207.03113*, 2022.

Vy Vo, Trung Le, Van Nguyen, He Zhao, Edwin V Bonilla, Gholamreza Haffari, and Dinh Phung. Feature-based learning for diverse and privacy-preserving counterfactual explanations. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2211–2222, 2023.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

Adam White and Artur d'Avila Garcez. Measurable counterfactual local explanations for any classifier. *arXiv preprint*, arXiv:1908.03020, 2019.

Muhammad Rehman Zafar and Naimul Mefraz Khan. Dlime: A deterministic local interpretable model-agnostic explanations approach for computer-aided diagnosis systems. *arXiv preprint*, arXiv:1906.10263, 2019.

Yujia Zhang, Kuangyan Song, Yiming Sun, Sarah Tan, and Madeleine Udell. " why should you trust my explanation?" understanding uncertainty in lime explanations. *arXiv preprint*, arXiv:1904.12991, 2019.

A Appendix

In the main paper, we presented the experiments using a neural network as a black-box model. Here, we show the results of experiments using a random forest of 1000 trees as a black-box model. The results are shown in Tables 6, 7, 8, and 9. They do not differ much from the results on the neural network black box model.

Table 6: Fidelity comparison for Random Forest as black-box model

	Factual				Counterf.
Dataset	LIME	MAPLE	BELLA	SHAP	BELLA
Auto MPG	1.63$\pm$0.430	**0.74**$\pm$0.218	1.36$\pm$0.421	**0.00**	4.02$\pm$0.950
Bike	322.72$\pm$16.83	**66.14**$\pm$5.420	176.42$\pm$10.58	**0.00**	362.99$\pm$21.30
Concrete	6.09$\pm$0.954	**2.36**$\pm$0.381	3.75$\pm$0.601	**0.00**	1.25$\pm$0.411
Servo	0.49$\pm$0.129	**0.16**$\pm$0.088	0.52$\pm$0.146	**0.00**	0.68$\pm$0.321
Electrical	**0.01**$\pm$0.001	**0.01**$\pm$0.000	**0.01**$\pm$0.00	**0.00**	**0.01**$\pm$0.000
Supercond.	28.40$\pm$1.470	**2.94**$\pm$0.409	5.41$\pm$0.499	**0.00**	43.47$\pm$1.513
White Wine	0.31$\pm$0.027	**0.16**$\pm$0.013	0.28$\pm$0.023	**0.00**	0.44$\pm$0.033
Real Estate	4.08$\pm$1.202	**3.16**$\pm$1.050	3.39$\pm$0.756	**0.00**	7.89$\pm$3.139
Wind	1.49$\pm$0.084	**0.55**$\pm$0.036	1.01$\pm$0.060	**0.00**	4.61$\pm$0.180
CPU Activity	11.66$\pm$1.030	**0.71**$\pm$0.101	1.44$\pm$0.238	**0.00**	0.88$\pm$0.058
Echocard.	3.51$\pm$0.113	**1.74**$\pm$0.069	3.17$\pm$0.121	**0.00**	12.31$\pm$0.402
Iranian Churn	146.68$\pm$20.73	**10.92**$\pm$2.335	14.28$\pm$2.543	**0.00**	150.77$\pm$20.35
Norm. avg.	0.07$\pm$0.008	0.02$\pm$0.004	0.04$\pm$0.005	0.00	0.09$\pm$0.012

Table 7: Generality comparison (% - larger is better)

Dataset	LIME	SHAP	MAPLE	BELLA
Auto MPG	4.25$\pm$2.653	0.00	43.91$\pm$3.493	**77.05**$\pm$9.229
Bike	1.30$\pm$0.225	0.00	5.48$\pm$0.089	**39.99**$\pm$2.392
Concrete	0.33$\pm$0.168	0.00	**31.18**$\pm$1.444	23.26$\pm$4.079
Servo	4.63$\pm$1.900	0.00	79.57$\pm$5.023	**81.69**$\pm$12.721
Electrical	0.01$\pm$0.001	0.00	**17.12**$\pm$0.305	16.89$\pm$1.882
Supercond.	0.01$\pm$0.159	0.00	14.93$\pm$0.439	**53.55**$\pm$2.386
White Wine	0.68$\pm$0.245	0.00	18.44$\pm$0.306	**65.31**$\pm$2.879
Real Estate	2.52$\pm$0.968	0.00	49.42$\pm$3.936	**55.38**$\pm$12.020
Wind	0.37$\pm$0.057	0.00	12.41$\pm$0.176	**99.97**$\pm$0.014
CPU Activity	0.75$\pm$0.146	0.00	9.54$\pm$0.215	**51.63**$\pm$1.570
Echocard.	0.31$\pm$0.040	0.00	5.90$\pm$0.081	**86.40**$\pm$1.080
Iranian Churn	1.5$\pm$0.216	0.00	11.91$\pm$0.429	**12.49**$\pm$1.843
Average	1.18$\pm$0.565	0.00	24.98$\pm$1.328	**55.21**$\pm$4.341

Table 8: Simplicity comparison (smaller values are better)

Dataset	SHAP	MAPLE	BELLA/LIME
Auto MPG	8.00$\pm$0.000	7.90$\pm$0.097	**3.90**$\pm$0.382
Bike	12.00$\pm$0.032	12.57$\pm$0.080	**7.43**$\pm$0.303
Concrete	8.00$\pm$0.000	7.00$\pm$0.000	**5.40**$\pm$0.301
Servo	10.12$\pm$0.845	15.94$\pm$0.125	**6.76**$\pm$1.068
Electrical	12.00$\pm$0.000	12.00$\pm$0.000	**8.06**$\pm$0.201
Supercond.	48.90$\pm$1.159	80.00$\pm$0.007	**15.50**$\pm$0.344
White Wine	11.00$\pm$0.000	10.00$\pm$0.000	**6.70**$\pm$0.291
Real Estate	5.00$\pm$0.000	4.00$\pm$0.000	**3.76**$\pm$0.256
Wind	13.63$\pm$0.048	13.00$\pm$0.000	**7.82**$\pm$0.155
CPU Activity	12.00$\pm$0.000	11.00$\pm$0.000	**10.35**$\pm$0.523
Echocardiogram	9.00$\pm$0.000	7.48$\pm$0.025	**5.65**$\pm$0.145
Iranian Churn	9.19$\pm$0.043	9.44$\pm$0.063	**5.56**$\pm$0.176
Norm. Avg.	0.90$\pm$0.006	0.89$\pm$0.003	**0.57**$\pm$0.022

Table 9: Robustness comparison (0 to 1 – larger is better)

Dataset	LIME	SHAP	MAPLE	BELLA
Auto MPG	**0.87**±0.017	0.67±0.044	0.65±0.060	0.70±0.028
Bike	0.77±0.004	0.66±0.005	0.51±0.063	**0.79**±0.009
Concrete	**0.76**±0.020	0.65±0.019	0.70±0.050	0.69±0.035
Servo	**0.90**±0.019	0.79±0.038	0.52±0.108	0.75±0.030
Electrical	**0.82**±0.002	0.50±0.004	0.59±0.028	0.81±0.006
Supercond.	0.76±0.006	0.85±0.005	0.49±0.033	**0.80**±0.011
White Wine	0.62±0.007	0.51±0.007	0.59±0.042	**0.77**±0.013
Real Estate	0.63±0.040	0.69±0.038	0.66±0.057	**0.85**±0.060
Wind	0.71±0.006	0.63±0.006	0.61±0.027	**0.98**±0.002
CPU Activity	0.52±0.005	0.69±0.008	0.65±0.040	**0.84**±0.006
Echocardiogram	0.81±0.004	0.52±0.002	0.46±0.037	**0.99**±0.011
Iranian Churn	0.75±0.017	**0.79**±0.011	0.65±0.055	0.78±0.012
Average	0.74±0.012	0.66±0.016	0.59±0.050	**0.81**±0.019