Generalized Oversampling for Learning from Imbalanced datasets and Associated Theory: Application in Regression

Anonymous authors
Paper under double-blind review

Abstract

In supervised learning, it is quite frequent to be confronted with real imbalanced datasets. This situation leads to a learning difficulty for standard algorithms. Research and solutions in imbalanced learning have mainly focused on classification tasks. Despite its importance, very few solutions exist for imbalanced regression. In this paper, we propose a data augmentation procedure, the GOLIATH algorithm, based on kernel density estimates and especially dedicated to the problem of imbalanced data. This general approach encompasses two large families of synthetic oversampling: those based on perturbations, such as Gaussian Noise, and those based on interpolations, such as SMOTE. It also provides an explicit form of such machine learning algorithms. New synthetic data generators are deduced. We apply GOLIATH in imbalanced regression combining such generator procedures with a new wild-bootstrap resampling technique for the target values. We evaluate the performance of the GOLIATH algorithm in imbalanced regression where we compare our approach with state-of-the-art techniques.

1 Introduction

Many real-world forecasting problems are based on predictive models in a supervised learning framework and standard algorithms can fail when the target variable is too skewed. Learning from imbalanced data concerns many problems with numerous applications in different fields (Krawczyk (2016), Fernández et al. (2018a)). The major part of such works concerns imbalanced classification (see for instance Buda et al. (2018), Cao et al. (2019), Cui et al. (2019), Huang et al. (2016), Yang & Xu (2020), Branco et al. (2016b) where many solutions propose a pre-processing strategy especially for the generation of new synthetic data. A large part of such existing methods consist in adapting the well know SMOTE algorithm (Fernández et al. (2018b)). Very few works have addressed the problem of imbalanced regression although many important real-world applications in different fields such as economy, meteorology, or insurance. As for imbalanced classification, some applications focus on predicting rare and extreme values, which can be of great interest. In the literature, the imbalanced regression corresponds to *the correct prediction of rare extreme values of a continuous target variable* Fernández et al. (2018b) but, contrary to the classification tasks, there is no level to quantify the imbalance and the labels are continuous. Unlike in a classification context, learning from imbalanced dataset for regression tasks leads to two additional problems: i) the definition of the imbalanced phenomenon and ii) the identification of the observations that are considered as minority.

In this paper we propose a very general method, which we shall call GOLIATH (for Generalized Oversampling for Learning from Imbalanced datasets and Associated THeory) to deal with the imbalanced regression problem. The first step of GOLIATH is a synthetic covariates generation based on kernel density estimators. The second step of GOLIATH is concerned with the imbalanced regression: a new method based on a wild-bootstrap procedure is proposed for generating target values given the synthetic covariates. GOLIATH is then a two-step algorithm. In this paper, we concentrate on tabular data rather than images because many applications rely on structured data, and there are still very few solutions available to address them. Our main contributions can be summarized as follows:

i) Providing a unified statistical expression of existing data augmentation algorithms, including the popular SMOTE;

ii) Deducing new synthetic data oversampling methods using the large and flexible expression of GOLIATH;

iii) Proposing innovative methodology to generate the target variable for dealing with imbalanced regression: conditioning on the newly generated covariates and avoiding discretizing its support.

iv) Proposing a new way to generate synthetic data providing a solution to the main challenge of synthetic data generation: avoiding overfitting and the introduction of bias.

The paper is organized as follows. In Section 3 we give a general form of our data augmentation procedure corresponding to the first step of GOLIATH. We study some standard perturbation and interpolation methods that are included in this approach, such as SMOTE and Gaussian Noise. In Section 5 we develop the theory to obtain new generators. In Section 6 we will look more closely at the imbalanced regression, corresponding to the second step of GOLIATH. Numerical results on several applications are presented in Section 7. Finally, we discuss the method proposed in Section 7.3.

2 Related work and main differences with Goliath

Most of the works in imbalanced regression have proposed to binarize the problem with a relevant function and an associate threshold Torgo & Ribeiro (2007) in order to adapt some imbalanced classification solutions Torgo et al. (2013), Branco et al. (2017), Branco et al. (2019), Ribeiro & Moniz (2020), Song et al. (2022), Camacho et al. (2022). This methodology presents the disadvantage of dividing the continuous distribution of the target variable into binary classes and therefore involves a loss of information. More recently, other methods have emerged by using deep learning approaches, for dealing with images, such as Sen et al. (2023), Ding et al. (2022), Gong et al. (2022) or Wang & Wang (2024). For instance Yang et al. (2021) proposed to use kernel density estimates to improve learning from imbalanced data with continuous targets. Like the previous works, this proposal suggests dividing the target variable support into B groups that involve a loss of information. However, the techniques proposed in this context rely on deep learning, which is effective for images but less so for tabular data, which is our application framework here.

In regression, the continuous nature of Y introduces two challenges: i) unlike classification where it is immediate to detect imbalance by comparing classes, measuring imbalance in regression is challenging because the data is continuous. GOLIATH provides a solution by using the inverse of the kernel density to determine the weights associated with Y; ii) unlike classification, where the labels of Y remain unchanged during synthetic data creation, in regression it is necessary to generate new and relevant values for the target variable. In this way, GOLIATH combines a two-step procedure: the first generates X and the second deduces the generation of Y using an innovative procedure based on a wild bootstrap.

3 A New Kernel-Based Oversampling Formulation

4 General Formulation of GOLIATH

We consider a sequence of observations $\{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)\}$, which are realizations of n iid random variables $(\boldsymbol{X}, Y)$, where the target variable Y is univariate and the covariate $\boldsymbol{X}$ is a p-dimensional random vector. The components of $\boldsymbol{X} = (X_1, \cdots, X_p)$ are supposed to be continuous or discrete and Y is supposed to be quantitative.

Write $\tilde{\boldsymbol{x}} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}$ the set of all observations. We call GOLIATH the generalized oversampling procedure based on the form of the following weighted kernel density estimate:

$$g_{\boldsymbol{X}^*}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \sum_{i \in \mathcal{I}} \omega_i K_i(\boldsymbol{x}^*, \tilde{\boldsymbol{x}}), \tag{1}$$

where $(K_i)_{i \in \mathcal{I}}$ is a collection of kernels, $(\omega_i)_{i \in \mathcal{I}}$ is a sequence of positive weights with $\sum_{i \in \mathcal{I}} \omega_i = 1$, and $\mathcal{I}$ represents a subset of $\{1, 2, \cdots, n\}$. Here the index * stands for the synthetic data. In (1) we propose a general form for the conditional density for the synthetic data generators. The objective is to use the flexibility of the kernels to estimate the density of covariates in order to obtain synthetic data that reflects the distribution of the observations.

We can show that (1) generalizes perturbation-based and interpolation-based synthetic data oversampling. We give an illustration with the basic algorithms ROSE, Gaussian Noise and SMOTE in Subsections 4.1 and 4.2 where we demonstrate that these methods are particular cases of the generalized form (1), with corresponding parameters summarized in Appendix A.2. Several existing methods can be rewritten in the form (1) and we give some illustrations in Appendix A.3. In Section 5 we will show that some particularly interesting new methods can be derived from the generic form (1) and we will compare some of them to current competitors in the imbalanced regression context.

Remark **1.** *The generators in (1) can be considered as smoothed bootstrap methods (see Silverman & Young (1987), Hall et al. (1989), De Angelis & Young (1992)). Smoothed bootstrap consists in drawing samples from kernel density estimators of the distribution. It can be decomposed into two steps: first, a seed is randomly drawn and second, a random noise from the kernel density estimator is added to obtain a new sample. In the form (1), the first step is represented by the drawing weight ω_i and the second by the kernel $K_i(x)$. Convergence properties of smoothed bootstrap are studied in De Martini & Rapallo (2008) and Falk & Reiss (1989). They proved the consistency of the smoothed bootstrap with classical multivariate kernel estimator and more specifically the convergence in Mallows metric. As described by the authors, the smoothed bootstrap provides better performances than a classical bootstrap when a proper choice of smoothing parameters is used. Other works have focused on the consistency of the multivariate kernel density estimate and proposed a relevant bandwidth matrix, see for instance Silverman (1986), Scott (2015) and Duong & Hazelton (2005).*

4.1 Rewriting Interpolation Approaches

As presented in Fernández et al. (2018b), the Synthetic Minority Oversampling Technique (SMOTE) Chawla et al. (2002) is considered a "de facto" standard for learning from imbalanced data and has inspired a large number of methods to handle the issue of class imbalance. It is also one of the first techniques adapted to imbalanced target values in regression with Torgo et al. (2013). SMOTE algorithm can be summarized as follows: at each step of the data augmentation procedure[1], an observation is randomly selected, which we shall call *a seed*. We will denote by S the random variable indicating the index of the seed, that is $S = i$ if the ith observation has been selected, and we denote by $S(1), \cdots, S(k)$ the k nearest neighbors (k-nn) of $\boldsymbol{x}_S$. Given S, a neighbor denoted by $N_k(S)$ is randomly chosen among $S(1), \cdots, S(k)$. The new data is generated by linear interpolation between S and $N_k(S)$. We have $\mathbb{P}(S = i) = \frac{1}{n}$, $\forall i = 1, \cdots, n$, and $\mathbb{P}(N_k(S) = S(\ell)) = \frac{1}{k}$, $\forall \ell = 1, \cdots, k$. Finally, writing $\boldsymbol{X}^*$ the synthetic random vector we have $\boldsymbol{X}^* = \lambda \boldsymbol{x}_S + (1-\lambda) N_k(S)$, with λ uniformly distributed $\mathcal{U}([0; 1])$.

To show that this approach is a particular case of (1) we proceed in three steps:

1 Conditionally to S and $N_k(S)$, the jth component of $\boldsymbol{X}^*$ is generated by a uniform distribution

$$g^{SMOTE}_{\boldsymbol{X}^*}\big(\boldsymbol{x}^*_j|\tilde{\boldsymbol{x}}, S = i, N_k(S) = \boldsymbol{x}_i(\ell)\big) = \frac{\mathbb{1}_{[\boldsymbol{x}_{ij}, \boldsymbol{x}_{ij}(\ell)]}(\boldsymbol{x}^*_j)}{|\boldsymbol{x}_{ij}(\ell) - \boldsymbol{x}_{ij}|} = \frac{\mathbb{1}_{[0, \boldsymbol{x}_{ij}(\ell) - \boldsymbol{x}_{ij}]}(\boldsymbol{x}^*_j - \boldsymbol{x}_{ij})}{|\boldsymbol{x}_{ij}(\ell) - \boldsymbol{x}_{ij}|},$$

where $\boldsymbol{x}_s(\ell)$ is the ℓth nearest neighbors of $\boldsymbol{x}_s$. Each component of $\boldsymbol{X}^*$ is drawn by the same uniform variable, that is $\boldsymbol{X}^*_j = \lambda \boldsymbol{x}_{Sj} + (1-\lambda) N_k(S)_j$ for $j = 1, \cdots, p$, and by abuse of notation we write the multivariate generating density as follows:

$$g^{SMOTE}_{\boldsymbol{X}^*}\big(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}, S = i, N_k(S) = \boldsymbol{x}_i(\ell)\big) = \frac{\mathbb{1}_{[0, \boldsymbol{x}_i(\ell) - \boldsymbol{x}_i]}(\boldsymbol{x}^* - \boldsymbol{x}_i)}{|\boldsymbol{x}_i(\ell) - \boldsymbol{x}_i|}.$$

2 Conditionally to S, $\boldsymbol{X}^*$ is generated according to a uniform mixture model (UMM) on the segments between $\boldsymbol{x}_S$ and its k-nn. The same mixture component is used for each component giving

$$g^{SMOTE}_{\boldsymbol{X}^*}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}, S = i) = \frac{1}{k} \sum_{\ell=1}^{k} \frac{\mathbb{1}_{[0, \boldsymbol{x}_i(\ell) - \boldsymbol{x}_i]}(\boldsymbol{x}^* - \boldsymbol{x}_i)}{|\boldsymbol{x}_i(\ell) - \boldsymbol{x}_i|}.$$

[1]In the original version of SMOTE, the seed is drawn successively with a loop and not randomly. These two ways are very close when the generated sample size is large

3 More generally, $\boldsymbol{X}^*$ is generated according to a mixture of UMM as follows:
since $\mathbb{P}(S=i)=\frac{1}{n}$, we have

$$g_{\boldsymbol{X}^*}^{SMOTE}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \sum_{i=1}^{n} g^{SMOTE}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}, S=i) \times \frac{1}{n} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{k}\sum_{\ell=1}^{k}\frac{\mathbb{1}_{[0,\boldsymbol{x}_i(\ell)-\boldsymbol{x}_i]}(\boldsymbol{x}^*-\boldsymbol{x}_i)}{|\boldsymbol{x}_i(\ell)-\boldsymbol{x}_i|}$$
$$= \frac{1}{n}\sum_{i=1}^{n} K_i^{SMOTE}(\boldsymbol{x}^*, \boldsymbol{x})$$

We finally obtain the form (1) with $\mathcal{I}=[0,n]$, $\omega_i = 1/n$ and $K_i(\boldsymbol{x}^*, \boldsymbol{x}) = K_i^{SMOTE}(\boldsymbol{x}^*, \boldsymbol{x})$. This new writing represents the conditional SMOTE density given the observation $\tilde{\boldsymbol{x}}$. We can relate this expression to the work of Elreedy et al. (2023) in which the authors give an expression of the unconditional SMOTE density, that is integrating the distribution over $\tilde{\boldsymbol{x}}$ in a context of class minority. Other methods derived from SMOTE can be recovered by (1) (see Appendix A.3).

4.2 Rewriting Perturbation Approaches

We illustrate (1) by recovering two classical data augmentation procedures, ROSE and Gaussian Noise (GN), as follows:

- At each step of the ROSE algorithm (see Menardi & Torelli (2014)) the seed S is selected randomly. Given S a synthetic data is generated with a multivariate density

$$g_{\boldsymbol{X}^*}^{ROSE}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}, S=i) = K_{H_n}^{ROSE}(\boldsymbol{x}^*-\boldsymbol{x}_i) = \frac{1}{|H_n|^{1/2}}K(H_n^{-1/2}(\boldsymbol{x}^*-\boldsymbol{x}_i))),$$

  where K denotes the multivariate Gaussian kernel and $H_n = diag(h_1,...,h_p)$ is the bandwidth matrix proposed by Bowman & Azzalini (1999), with $h_q = (\frac{4}{(p+2)n})^{1/(p+4)}\widehat{\sigma}_q, q=1,...,p$. Finally, a synthetic random variable $\mathbf{X}^*$ is generated with the density

$$g_{\boldsymbol{X}^*}^{ROSE}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \frac{1}{n}\sum_{i=1}^{n} K_{H_n}^{ROSE}(\boldsymbol{x}^*-\boldsymbol{x}_i) = \sum_{i=1}^{n}\omega_i K_{H_n}^{ROSE}(\boldsymbol{x}^*-\boldsymbol{x}_i)).$$

- Similarly to ROSE, at each step of the Gaussian Noise algorithm (see Lee & Sauchi (2000)) a seed is selected and synthetic data is generated. Finally, the generating multivariate density has the form

$$g_{\boldsymbol{X}^*}^{GN}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \frac{1}{n}\sum_{i=1}^{n} K_{H_n}^{GN}(\boldsymbol{x}^*-\boldsymbol{x}_i) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|H_n|^{1/2}}K(H_n^{-1/2}(\boldsymbol{x}-\boldsymbol{x}_i)),$$

  where $H_n^{GN} = diag(h_1,...,h_p)$, $h_q = \sigma_{noise}\widehat{\sigma}_q, q=1,...,p$.

Both cases are particular cases of (1) with $\omega_i = \frac{1}{n}$ and $K_i(\tilde{\boldsymbol{x}}, \boldsymbol{x}) = K_{H_n}(\boldsymbol{x}-\boldsymbol{x}_i)$, i.e. the same Gaussian kernel for all observations but with a different bandwidth matrix.

4.3 Global criticism

Although there are many extensions of SMOTE or ROSE and Gaussian Noise, such techniques suffer from some drawbacks. For the interpolations techniques, the directions in the data space are limited and deterministic because they depend only on the k-nn (nearest neighbors). Moreover, the distance from the seed is also limited because the new sample is on the segment with the drawn nearest neighbor. For the perturbation techniques, the directions in the data space are randomly generated and so they can more explore the space. The distance between the new sample and the seed is also unbounded. However, the directions are randomly chosen and do not respect the correlation between the data and their support and the correlations between variables.

5 New Kernel-Based Methods

5.1 Generalized Interpolation Approaches

A general form We propose a particular form of (1) which generalizes the SMOTE algorithm as follows:

$$g^{int}_{\boldsymbol{X}^*}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \sum_{i\in\mathcal{I}} \omega_i K_i(\boldsymbol{x}^*, \boldsymbol{x}) = \sum_{i\in\mathcal{I}} \omega_i \sum_{\ell\in\mathcal{J}_i} \pi_{\ell|i}\, g^{int}_{i,\ell}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}})$$

where $g^{int}_{i,\ell}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}})$ is an interpolation function on $[\boldsymbol{x}_i, \boldsymbol{x}_i(\ell)]$, $\mathcal{J}_i$ denoting the set of k-nn associated to $\boldsymbol{x}_i$. SMOTE is then a particular case when $\omega_i = \frac{1}{n}$, $\pi_{\ell|i} = \frac{1}{k}$ and $g^{int}_{i,\ell}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \frac{\mathbb{1}_{[0;\boldsymbol{x}_i(\ell)-\boldsymbol{x}_i]}(\boldsymbol{x}^*-\boldsymbol{x}_i)}{|\boldsymbol{x}_i(\ell)-\boldsymbol{x}_i|}$ represents a uniform distribution between the vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_i(l)$.

Nearest Neighbors Smoothed Bootstrap Since the uniform distribution coincides with the Beta distribution with parameters $\alpha = \beta = 1$, a natural extension of SMOTE is to consider a general Beta distribution. We find the same idea in Yao et al. (2022) within another context. The very flexibility of the Beta distribution suggests us to propose $\boldsymbol{X}^* = \lambda \boldsymbol{x}_i + (1-\lambda)\boldsymbol{x}(\ell)_i$ for $i = 1, \cdots, n$, where λ follows a generalized Beta distribution. By abuse of notation, we get the following interpolation function:

$$g^{int}_{i,\ell}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}(\boldsymbol{x}^* - \boldsymbol{x}_i)^{\alpha-1}(\boldsymbol{x}_i(\ell) - \boldsymbol{x}^*)^{\beta-1}\mathbb{1}_{[0;\boldsymbol{x}_i(\ell)-\boldsymbol{x}_i]}.$$

Extended Nearest Neighbors Smoothed Bootstrap Finally, the previous methods based on interpolation are limited to the "seed - k-nn segments" and therefore do not reach all the data space. To avoid generating on a bounded or discrete support we propose to extend these approaches to any part of the support by adding a Gaussian distribution on the segment as follows:

$$g^{e-int}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \sum_{i\in\mathcal{I}} \omega_i \sum_{\ell\in\mathcal{J}_i} \pi_{\ell|i}\, g^{e-int}_{i,\ell}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}),$$

where the extended interpolation function is a Beta Gaussian mixture, that is, $g^{e-int}_{i,\ell}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}})$ is the density of a Gaussian distribution $N(\theta, \sigma^2)$ where θ is generated by $g^{int}_{i,\ell}(\boldsymbol{x}|\tilde{\boldsymbol{x}})$. To rely on the recent literature, we remark that SASYNO algorithm Gu et al. (2020) is a special case of this methodology. This extended version can be viewed as a hybrid method between interpolation and perturbation techniques. It provides a good compromise between the interpolation and perturbation approaches because it can generate in the whole data space as the perturbation approach i.e. constraint-free, but assigns a distribution to the directions towards the segments, that is orienting the perturbation toward the k-nn.

Remark **2.** *We tried to adapt the k-nearest neighbors density estimate (Biau & Devroye (2015)) that is a bandwidth-variable kernel (also called a balloon kernel) as a generator but its computation time is currently too high to be used.*

5.2 Generalized Perturbation Approaches

Classical Smoothed Bootstrap As the ROSE and GN techniques use a multivariate Gaussian kernel estimate with a diagonal bandwidth matrix, we can rewrite their associated generating density as follow:

$$g_{\boldsymbol{X}^*}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \sum_{i=1}^{n} \omega_i \prod_{j=1}^{p} K_{h_j}(x^*_j - x_{ij}) \tag{2}$$

with $K_{h_j}(u) = (2\pi)^{-1/2} h_j^{-1} e^{-\frac{1}{2h_j{}^2}u^2}$ the univariate gaussian kernel density estimator with smoothing parameter h_j. Such kernels are clearly not adapted for asymmetric, bounded or discrete variables. This remark is also true for the work of Yang et al. (2021) which uses some symmetric kernels to improve learning of imbalanced datasets.

Non-Classical Smoothed Bootstrap To fix the drawback of the classical kernel we extend (2) by adapting (1) to the support of $\boldsymbol{x}$, considering some non-classical kernels (we refer to some works handling the kernel density estimation for specific distributions inspired from Bouezmarni & Rombouts (2010), Someé (2015), Hayfield & Racine (2008), Chen (2000)). We rewrite (1) as

$$g_{\boldsymbol{X}^*}^{per}(\boldsymbol{x}^*|\tilde{\boldsymbol{x}}) = \sum_{i\in\mathcal{I}} \omega_i \prod_{j=1}^{p} K_{h_j}(x_j^*, x_{ij})$$

where $K_{h_j}(u, x)$ is a univariate kernel adapted to the nature of the jth variable and specifically defined on x as follows:

- Gaussian kernel for a variable defined on $\mathbb{R}$ (classical kernel):

$$K_h(u, x) = \frac{1}{h\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{u-x}{h})^2}.$$

- Binomial kernel for a discrete variable defined on $\mathbb{N}$:

$$K_h(u, x) = \frac{(x+1)!}{u!(x+1-u)!} \left(\frac{x+h}{x+1}\right)^u \left(\frac{1-h}{x+1}\right)^{x+1-u}.$$

- Gamma kernel for a positive asymmetric distribution defined on $[a, +\infty]$:

$$K_h(u, x) = \frac{u^{(x-a)/h}}{\Gamma(1+(x-a)/h)h^{1+(x-a)/h}} \exp\left(\frac{-u}{h}\right) \mathbb{1}_{[a,+\infty]}(u).$$

- Negative Gamma kernel for a negative asymmetric distribution defined on $[-\infty, b]$:

$$K_h(u, x) = \frac{u^{-(x-b)/h}}{\Gamma(1-(x-b)/h)h^{1-(x-b)/h}} \exp\left(\frac{-u}{h}\right) \mathbb{1}_{[-\infty,b]}(u).$$

- Beta kernel for a variable defined on $[0, 1]$:

$$K_h(u, x) = \frac{u^{x/h}(1-u)^{(1-x)/h}}{\mathcal{B}(\frac{x}{h}+1, \frac{1-x}{h}+1)} \mathbb{1}_{[0,1]}(u).$$

- Truncated Gaussian kernel for a variable defined on $[a, b]$:

$$K_h(u, x) = \frac{\alpha}{h\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{u-x}{h})^2} \mathbb{1}_{[a,b]}(u),$$
$$\alpha := \left(\int_a^b \frac{1}{h\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{u-x}{h})^2}\right)^{-1}$$

Note that if the Dirac kernel ($\mathbb{1}_{\boldsymbol{x}=\boldsymbol{x}_i}$) is used, we get the standard bootstrap: 1 includes also the simple oversampling. It is important to note that the GOLIATH algorithm uses an estimation of the smoothing parameter h provided by some specific R-package dedicated to the density estimation (for instance, it uses the Silverman estimation for the Gaussian kernel). Their estimates are based on properties of univariate consistency. Another technique to deal with skewed or heavy-tailed distributions is to apply a transformation of the data in order to use classical kernel density estimation (Charpentier & Flachaire (2015), Charpentier & Oulidi (2010)) but it necessitates proposing a relevant transformation which exceeds the scope of this paper.

Remark **3.** *The use of a diagonal bandwidth matrix in (2) does not take into account the correlation between variables. To improve this issue, we could consider a full (symmetric positive definite) smoothing matrix. In that case, we would use a multivariate kernel density estimate considering the correlation between the variables which would be optimal for generating data. However, the estimation of such a matrix is generally based on the covariance matrix which does not adequately capture non-linear correlations. In practice, it can be challenging, even inconsistent, to find a form of a multivariate kernel that adapts to all data and their support.*

5.3 Goliath Overview

The GOLIATH algorithm is summarized in Figure 1a. A cartography of GOLIATH is also given in Figure 1b. The algorithm gives the possibility to choose, with the "mode" parameter, the kind of the returned sample: a full synthetic dataset, an augmented one, or a mixed one. The mixed sample is constructed as follows: keep the original observation for the first occurrence of the seed and synthetic data for the next. This mode corresponds to performing an undersampling and an oversampling. More precisely, to preserve the maximum of information and avoid potential overfitting, we suggest to: keep the initial observation for its first drawing and generating synthetic data from it for the other drawing which is the "mix" mode in the GOLIATH algorithm. This technique theoretically helps in reducing bias (by keeping the real data) while avoiding overfitting (by not duplicating the same observations). More details on this option are given in Appendix B.

5.3.1 GOLIATH Overall Algorithm

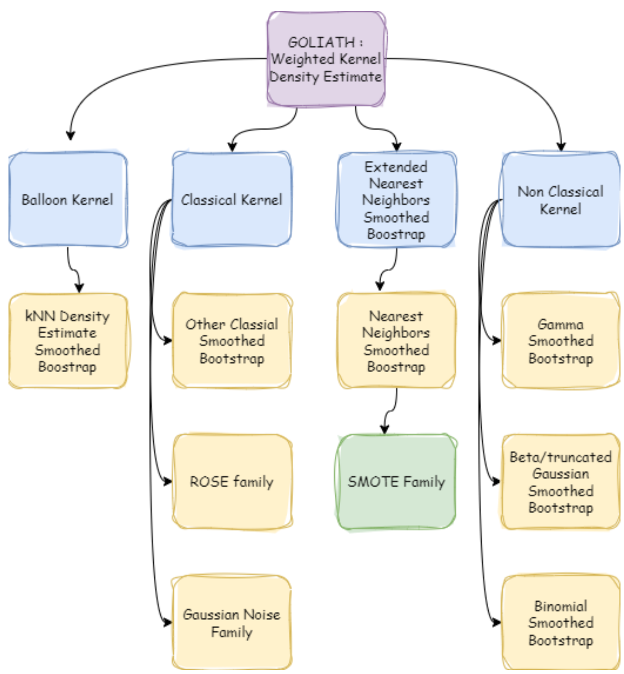
The GOLIATH algorithm is summarized in Figure 1a and Figure 1b presents its cartogrophy.

```
Input covariate X; target variable Y=null; mod="mix"; type; method-Y=1; clustering=F; seed s=null; components GMM m=n-row(X); weights w=rep(1,n-row(X)); synthetic data sample size N=n-row(X); parameter p)
Clustering    // Optional application of a clustering on the train
clust = Cluster(X,Y,clustering)    // clust=1 for all samples if clustering=F
Seed drawing    // weighted oversampling
if s = null then s = draw(X,N,w)
X generation    // Synthetic data generation for the covariates
if m < n then synth = GMM(X,Y,m,N)
  else for each c in clust:
    if type = "CSB" then synth = G-CSB(X,N,w, s, p)
    else if type = "NCSB" then synth = G-NCSB(X,N,w, s, p)

    else if type = "NNSB" then synth = G-NNSB(X,N,w, s, p)

    else if type = "eNNSB" then synth = G-eNNSB(X,N,w, s, p)
    Y generation    // Optional Synthetic Y generation
    if Y <> null then synth[,Y] = G-Y(X,Y,synth,s,method-Y, p)
  end For
Output
if m<n or mod = "synth" then return synth
else if mod = "augment" then return (X,Y) + synth
else return mix((X,Y),synth,s)
```

(a) GOLIATH algorithm



(b) GOLIATH cartography

Figure 1: GOLIATH algorithm and cartography

6 GOLIATH as a Solution for Imbalanced Regression

Using the first generator step of GOLIATH (methods proposed in 2.3.1 and 2.3.2) we can generate synthetic covariates X^*. We then have to generate the target variable Y given such X^*.

A non-parametric method for drawing weights We propose here to define the drawing weights ω_i in (1) as the inverse of the kernel density estimate for the target variable Y: the more isolated an observation is, the higher its drawing weight. To avoid giving disproportionate weights (typically too large a weight for extreme values), we can use a hyperparameter α as below. The weights are normalized to get a sum equal to 1. For $i = 1, \cdots, n$, the normalized drawing weight ω_i of observation i is thus defined as follows:

$$\omega_i := \frac{q_i}{\sum_j q_j}, \text{ with } q_i := \frac{1}{\widehat{f}_y(y_i)^\alpha}, \tag{3}$$

where $\widehat{f}_y$ denotes a convergent kernel density estimator.

In the following, we assume $\alpha = 1$.

Remark **4.** *With $\alpha = 1$, the generated random variable Y^* satisfying $\mathbb{P}(Y^* = y_i) = \omega_i$ is close to a uniform distribution. More precisely Stocksieker et al. (2023) generalized a result of Smith & Gelfand (1992) by showing that choosing $q_i = \frac{f_0(y_i)}{\widehat{f}_y(y_i)}$, with f_0 a target probability density function (pdf), then the pdf of Y^* converges to f_0. In our case f_0 is a uniform distribution and we can apply this result as soon as the support of Y is bounded.*

A similar idea, using the opposite, is explored in Steininger et al. (2021) where the authors propose to also use the kernel density estimate but with a different form. In Yang et al. (2021) the authors propose also using the inverse of the estimated label density to weight the loss function but apply a discretization of the target variable. The strength of our approach is to preserve the continuity of the distribution whereas most other works propose splitting it into bins that involve a loss of information. The kernel density is automatically adapted to the distribution of Y, as proposed in Subsection 5.2.

A new approach to generate the target variable The target variable is not generated in the same way as the covariate. Once the covariates are generated from our generator models, we propose to adapt a wild-bootstrap technique with synthetic features (Wu (1986))[2]. The classical *Wild-Bootstrap* involves uniformly drawing a prediction error ϵ_k to generate a new $y_i^* := \widehat{y}_i + \epsilon_k v_i$, where v_i is a random variable.

The generation of the target variable is performed as follows:

i) Train a Random Forest on the initial sample;
ii) For each seed $x_s \in \mathbf{x_S}$: predict target variable $\widehat{y}_s$ associated to x_s with the Random Forest; $\mathbf{x_S}$ being the original observations drawn at step 1.
iii) Obtain the distribution of the absolute residuals on the prediction of the seed: $y_s - \widehat{y}_s$ and draw a randomly residual ϵ_k;
iv) Generate the noise in the *Wild-Bootstrap* $v_s \sim \mathcal{N}(0, \sigma)$, σ being a parameter; v_s can also be set to 1 to avoid adding noise to the residual.
v) Generate a new y_s^* associated to the new synthetic x_s^* as follows:

$$y_s^* := y_s + |\epsilon_k| v_s \times sign(\widehat{y_s} - \widehat{y_s^*})$$

With $v_s = 1$, this form is close to the Wild Bootstrap version with the Rademacher distribution. The idea behind this proposition is to consider taking into account i) the prediction error and ii) the impact of the synthetic covariate on the target variable.

The choice of using a Random Forest is justified by its good predictive performance, its non-parametric nature, and the possibility of getting an error distribution for a given target variable value. It represents the second step of GOLIATH. Other interesting methods to generate Y are proposed in Appendix B.2 (but giving lower numerical performances on our illustration, as presented in Appendix 4a). The algorithm is detailed in Appendix B.2.

7 Application in Imbalanced Regression

To evaluate the performance of GOLIATH, we focus on the imbalanced regression context because of the natural capacity of the form (1) to handle continuous variables.

[2]The kernel regression (Nadaraya-Watson estimator) was also tested but not selected because its high computation time and poor performance

For each dataset, we construct a test sample as 10-30% of the initial dataset and use the weighting previously defined: the inverse of the kernel density estimate for the target variable Y. We construct an artificial imbalanced dataset from the remaining sample. More details on the protocol are given in Appendix 6. One of the objectives of our approach is to obtain better performance than the imbalanced train dataset. We compare our results to existing methods to deal with imbalanced regression from the *UBL* R-package, Branco et al. (2016a), classical oversampling, SMOTE, Gaussian Noise, SMOGN, WERCS and ADASYN from the python-package *ImbalancedLearningRegression* (Wu et al. (2023)). These techniques are used with their automatic relevance function and the same parameters as GOLIATH if any, in particular k for SMOTE and *pert* for the Gaussian Noise.

To avoid sampling effects and obtain a distribution of prediction errors we ran 10 train-test datasets. In the same way, to avoid getting results dependent on some learning algorithms we use 10 models from the *autoML of the H2O R-package* LeDell & Poirier (2020) among the following algorithms: Distributed Random Forest, Extremely Randomized Trees, Generalized Linear Model with regularization, Gradient Boosting Model, Extreme Gradient Boosting and a Fully-connected multi-layer artificial neural network. We present here the aggregated results of the models, a more detailed analysis is available in Appendix C and D.

We then compute the following metrics: RMSE and MAE and weighted-RMSE with our weighting function giving more importance to the rare values. Since the test sample is balanced on the target variable, we considered here the RMSE and MAE metrics as relevant to provide an overview of the average error across the whole target variable.

Remark **5.** *The basic methods (Gaussian Noise or SMOTE) are different from the UBL version because i) the generation of the target variable Y is realized with wild bootstrap and considers the new synthetic attributes and ii) the weights ω is defined for all samples while UBL uses a relevant function that divides the dataset into rare and frequent sets. Like the ROSE and Gaussian Noise algorithms, GOLIATH takes into account a parameter tuning the level noise for perturbation approaches (description in Appendix A). It is also possible to use a clustering (Gaussian Mixture Model) in GOLIATH in order to apply a generation by cluster. All datasets provided by GOLIATH in the applications were provided with the mod "mix". Note that the ROSE algorithm did not exist for imbalanced regression.*

7.1 Illustrative Application

In order to get a reference for predictive performance, we chose a balanced dataset from which we build an imbalanced train dataset. The dataset, named *SML2010 Data Set* is available on the Machine Learning repository UCI[3]. It is composed of 24 numeric attributes and 4137 instances. The target variable is the indoor temperature (we construct a unique target variable as the mean temperature of dinning-room and the temperature of the room). We train, with the autoML, the following train dataset:

- Reference values: Full sample (*FTrain*), Imbalanced (*Imb*)
- Benchmark: UBL-Oversampling (*UBL-OS*), UBL-SMOTE for regression (*UBL-SMOTE*), UBL-Gaussian Noise for regression (*UBL-GN*), UBL-SMOGN for regression (*UBL-SMOGN*), UBL-WERCS (*UBL-WERCS*), IRL-ADASYN (*IRL-OS*)
- GOLIATH (step 1): Oversampling (*G-OS*), Gaussian Noise (*G-GN*), Gaussian Noise with GMM-clustering (*G-GNwCl*), ROSE (*G-ROSE*), ROSE with a GMM-clustering (*G-ROSEwCl*), SMOTE (*G-SMOTE*), Non classical Smoothed Boostrap with contraints on the distributions (*G-NCSB*), Classical Smoothed Boostrap (*G-CSB*), Nearest Neighbors Smoothed Bootstrap, with Beta distribution, (*G-NNSB*), Nearest Neighbors Smoothed Bootstrap with k-NN weights proportionates to the distance from the seed (*G-NNSBw*), Extended Nearest Neighbors Smoothed Bootstrap (*G-eNNSB*).

Figure 2a shows the results for RMSE on the test sample. The weighted-RMSE and MAE metrics are shown in Appendix 7 and present similar results. We can observe that the GOLIATH algorithm presents an RMSE smaller than the imbalanced sample and than the benchmark techniques, whatever the generators. The GOLIATH-oversampling is comparable to the UBL-oversampling which confirms, on this dataset, the

[3] `https://archive.ics.uci.edu/ml/datasets/SML2010`

relevance of the weighting. The Non-Classical Smoothed Bootstrap (NCSB) is as efficient as the Classical Smoothed Bootstrap (CSB, ROSE, GN). However, it provides realistic values for the variables. We can see some examples of inconsistency in Appendix 4b. The results show that the clustering seems to improve the performance. The Nearest Neighbors Smoothed Bootstrap (NNSB) and Extended Nearest Neighbors Smoothed Bootstrap(eNNSB) outperform the original SMOTE. It is important to note that the different parameters (the number of nearest neighbors for interpolation techniques and the level of noise for perturbation techniques) are arbitrary and not optimized here. The heatmap in Figure 2b shows the robustness of the methods with their rank by run with respect to the RMSE. We can observe, based on the mean and standard deviation of the rank, that the Classic Smoothed Bootstrap, the Non-Classical Smoothed Bootstrap, and the Nearest Neighbors Smoothed Bootstrap are the best approaches here.

RMSE Rank	1	2	3	4	5	6	7	8	9	10	mean	sd
Ftrain	0,8	0,7	0,7	0,9	0,8	0,7	0,8	0,8	0,9	0,9	**0,8**	**0,1**
Imb	2,19	1,79	1,48	1,44	1,85	1,85	2,08	2,02	1,89	1,59	**1,82**	**0,25**
UBL-OS	2,21	1,96	1,69	1,82	2,12	1,83	2,38	2,22	2,07	1,85	**2,02**	**0,22**
UBL-SMOTE	1,87	1,85	1,60	1,53	1,68	1,97	2,12	2,13	1,85	1,77	**1,84**	**0,20**
UBL-GN	2,24	1,98	1,67	1,60	1,92	1,61	2,10	2,01	1,81	1,67	**1,86**	**0,22**
UBL-SMOGN	2,17	1,81	1,74	1,67	1,92	1,96	2,20	2,17	1,94	1,52	**1,91**	**0,23**
UBL-WERCS	2,43	1,87	1,80	1,59	2,08	1,64	2,24	2,09	2,13	1,86	**1,97**	**0,27**
IRL-ADASYN	2,63	2,26	2,17	2,03	2,58	2,31	2,50	2,57	2,44	2,33	**2,38**	**0,20**
G-OS	2,11	1,83	1,67	1,71	1,86	2,10	2,16	2,21	2,13	1,93	**1,97**	**0,20**
G-GN	2,00	1,82	1,22	1,48	1,76	1,49	1,57	1,92	1,61	1,66	**1,65**	**0,23**
G-GNwCl	1,65	1,61	1,56	1,37	1,54	1,99	1,91	1,78	1,65	1,48	**1,65**	**0,19**
G-ROSE	1,60	1,73	1,69	1,35	1,57	1,47	1,68	1,93	1,71	1,60	**1,63**	**0,16**
G-ROSEwCl	1,84	1,60	1,29	1,48	1,43	1,79	1,77	2,01	1,77	1,49	**1,65**	**0,22**
G-SMOTE	1,95	1,80	1,61	1,54	1,63	1,73	2,25	2,67	1,30	1,93	**1,84**	**0,39**
G-NCSB	2,19	1,77	1,54	1,33	1,44	1,63	1,70	2,05	1,78	1,66	**1,71**	**0,26**
G-CSB	1,69	1,68	1,55	1,30	1,50	1,82	1,78	2,00	1,72	1,52	**1,66**	**0,20**
G-NNSB	1,67	1,56	1,57	1,30	1,77	1,54	1,79	1,29	1,45	1,60	**1,55**	**0,17**
G-NNSBw	1,62	1,97	1,46	1,46	1,64	1,83	1,54	1,90	1,64	1,74	**1,68**	**0,18**
G-eNNSB	2,20	1,79	1,45	1,15	1,29	1,63	1,73	1,86	1,46	1,57	**1,61**	**0,30**

(a) RMSE Heatmap

RMSE Rank	1	2	3	4	5	6	7	8	9	10	mean	sd
Imb	13	8	5	7	12	13	10	10	13	6	**10**	**3**
UBL-OS	15	15	15	17	17	12	17	16	15	14	**15**	**2**
UBL-SMOTE	7	13	10	11	9	15	12	13	12	13	**12**	**2**
UBL-GN	16	17	13	14	15	4	11	9	11	11	**12**	**4**
UBL-SMOGN	11	10	16	15	15	14	14	14	14	4	**13**	**4**
UBL-WERCS	17	14	17	13	16	7	15	12	17	15	**14**	**3**
IRL-ADASYN	18	18	18	18	18	18	18	17	18	18	**18**	**0**
G-OS	10	12	13	16	13	17	13	15	17	17	**14**	**2**
G-GN	9	11	1	10	10	2	2	5	4	10	**6**	**4**
G-GNwCl	3	3	8	6	5	16	9	2	6	1	**6**	**4**
G-ROSE	1	5	15	5	6	1	3	6	7	8	**6**	**4**
G-ROSEwCl	6	2	2	10	2	9	6	9	9	2	**6**	**3**
G-SMOTE	8	9	11	12	7	8	16	18	1	17	**11**	**5**
G-NCSB	13	6	6	4	3	6	4	11	10	10	**7**	**3**
G-CSB	5	4	7	3	4	10	7	7	8	4	**6**	**2**
G-NNSB	4	1	9	3	11	3	8	1	2	8	**5**	**4**
G-NNSBw	2	16	4	8	8	12	1	4	5	12	**7**	**5**
G-eNNSB	14	8	3	1	1	6	5	3	3	5	**5**	**4**

(b) RMSE Ranking

Figure 2: Numerical simulation. RMSE values and ranking for the full sample, the imbalanced sample, 6 competitors, and GOLIATH associated with 11 different generating methods

The RMSE-rank represents the ranking of approaches according to the RMSE for a run: rank 1 corresponds to the training dataset that offers the smallest RMSE on the test sample. We also compared the results using the R-package *IRon: Solving Imbalanced Regression Tasks* [4], a useful and relevant package specific to Imbalanced Regression based on Ribeiro & Moniz (2020). The results in Appendix C.6 demonstrate that GOLIATH outperforms UBL approaches, even when considering their performance metrics (weighted MSE, weighted MAE, and SERA).

7.2 Imbalanced Regression Applications

We test our approach on several real data set from a repository provided as a benchmark for imbalanced regression problems[5] and presented in Branco et al. (2019) (descriptions in Appendix D). Figures 3a and 3b present RMSE gain (wrt the imbalanced dataset) and the median of the RMSE ranking. We can observe on these datasets that the GOLIATH algorithm empirically outperforms the state-of-the-art techniques, especially the Non-Classical Smoothed Bootstrap and the Extended Nearest Neighbors Smoothed Bootstrap.

We can see on these several applications, with several runs, several learning algorithms, and several performance metrics that the GOLIATH approach seems relevant to deal with imbalanced regression. In general GOLIATH gives better results, especially when it is combined with and extended nearest neighbors smoothed bootstrap in its first step of covariates generation.

7.3 Discussion

Based on all our numerical evidences, we strongly recommend the use of GOLIATH when step 1 combines a smoothed boostrap with nearest neighbors. The five versions of GOLIATH involving bootstrap procedures appear to be the best performing and most stable in terms of numerical results (both RMSE gain and rank).

[4] `https://cran.r-project.org/web/packages/IRon/IRon.pdf`
[5] `https://paobranco.github.io/DataSets-IR/`

RMSE gain	NO2	cpuSm	Boston	Bank8FM	Abalone
UBL-OS	-3%	2%	-7%	63%	0%
UBL-SMOTE	-10%	-2%	-12%	27%	-4%
UBL-GN	-8%	-5%	-7%	57%	-3%
UBL-SMOGN	-10%	-3%	-10%	29%	-3%
UBL-WERCS	-4%	3%	-1%	57%	-1%
IRL-ADASYN	10%	22%	-1%	69%	NA
G-OS	-1%	-2%	-3%	57%	-3%
G-GN	-11%	-18%	-21%	0%	-9%
G-GNwCl	-10%	-18%	-14%	13%	-6%
G-ROSE	-9%	-17%	-17%	-6%	-9%
G-ROSEwCl	-9%	-19%	-21%	0%	-8%
G-SMOTE	-6%	-2%	-6%	54%	-11%
G-NCSB	-9%	-23%	-23%	-6%	-9%
G-CSB	-12%	-20%	-17%	0%	-9%
G-NNSB	-13%	-11%	-12%	31%	-14%
G-NNSBw	-15%	-11%	-14%	35%	-13%
G-eNNSB	-15%	-7%	-19%	-29%	-21%

(a) RMSE-gain

RMSE rank	NO2	cpuSm	Boston	Bank8FM	Abalone
UBL-OS	14,0	14,5	13,0	17,0	16,5
UBL-SMOTE	10,0	11,5	11,0	11,0	12,0
UBL-GN	10,5	9,5	11,5	15,5	13,0
UBL-SMOGN	8,0	12,0	11,5	12,0	12,0
UBL-WERCS	14,0	16,0	15,5	16,5	15,0
IRL-ADASYN	18,0	18,0	16,0	18,0	NA
G-OS	16,0	14,0	14,5	15,5	13,5
G-GN	6,5	5,5	2,5	6,0	6,5
G-GNwCl	8,5	5,0	6,5	8,0	9,5
G-ROSE	8,5	4,0	4,0	4,5	7,0
G-ROSEwCl	10,0	4,0	3,0	8,0	8,0
G-SMOTE	13,5	11,5	12,5	15,5	4,5
G-NCSB	8,0	2,0	4,0	5,0	5,5
G-CSB	6,5	3,5	6,0	5,0	6,0
G-NNSB	6,0	8,5	8,0	12,0	2,5
G-NNSBw	3,0	7,0	6,5	12,0	3,5
G-eNNSB	2,0	10,0	5,5	1,5	1,0

(b) Median of the RMSE-rank

Figure 3: Datasets. RMSE values and ranking for 6 competitors, and GOLIATH associated with 11 different generating methods

8 Discussion and Perspectives

GOLIATH is an algorithm gathering two large families of synthetic data oversampling. Many methods can be rewritten as particular cases of it. This approach gets the advantage to obtain a general form for the generator which is based both on the theoretical foundations of kernel estimators and classical smoothed bootstrap techniques. It provides a general expression for the conditional density of the generator. The use of well-chosen kernels makes it possible to take into account the nature of the covariates: continuous, discrete, totally or partially bounded. Our approach generalizes the SMOTE algorithm by providing weights and flexible densities for interpolation. We also extend this technique to wider support than that of the observations by combining interpolation and perturbation approaches. Numerical applications in imbalanced regression models demonstrate that GOLIATH and its variants are very competitive, especially when the generator used in step 1 is the extended nearest neighbors smoothed bootstrap.

The weights ω_i (and $\pi_{j|i}$ in the interpolation case) offer a large flexibility. For instance, it is possible to handle classification tasks by conditioning with the minority class. We could deal with multi-class classification too. It is also possible to combine some extensions of SMOTE that propose to focus on specific samples in the synthetic data generation (as ADASYN) with a kernel approach in order to perform the methodology.

As a perspective, a natural extension of this work is to automate the choice of the kernel estimators, the weights, as well as some parameters according to the data. For example by defining a weights function for the nearest neighbor instead of defining the parameter k. Indeed, the parameter k is sometimes unsuitable and we could suggest a dynamic weighting depending on the neighborhood. It is also possible to define a kernel according to the neighborhood into the same dataset. For instance, an interpolation approach is favored within clusters when neighboring points are considered close to the observation. On the other hand, a perturbation approach is preferred when the observation appears isolated. Finally, non-standard kernels enable handling specific distributions such as bounded or discrete ones.

We also could define ω_i in order to generate a target distribution as done in Stocksieker et al. (2023). Finally, the perturbation-based approaches, based on kernel density estimators, may find it challenging to accurately capture dependencies between variables. The interpolation approaches consider it but the generation is limited to the segments. The extended-SMOTE proposes a first solution. GOLIATH proposes also an innovative method to generate Y based on the generated X, regardless of the generator used. It would also be interesting and potentially effective to use multiple generators and capitalize on the strengths of each. The generators could be applied locally based on the data characteristics.

Another research direction would be to better consider i) correlations between variables while respecting their definition domain and ii) mixed data. Finally, it could be interesting to test GOLIATH on image datasets, by combining it with a deep-learning model (Deep Imbalanced Regression framework).

References

Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004.

Gérard Biau and Luc Devroye. *Lectures on the nearest neighbor method*, volume 246. Springer, 2015.

Taoufik Bouezmarni and Jeroen VK Rombouts. Nonparametric density estimation for multivariate bounded data. *Journal of Statistical Planning and Inference*, 140(1):139–152, 2010.

Adrian W. Bowman and Adelchi Azzalini. Applied smoothing techniques for data analysis : the kernel approach with s-plus illustrations. *Journal of the American Statistical Association*, 94:982, 1999.

Paula Branco, Rita P Ribeiro, and Luis Torgo. Ubl: an r package for utility-based learning. *arXiv preprint arXiv:1604.08079*, 2016a.

Paula Branco, Luís Torgo, and Rita P Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM computing surveys (CSUR)*, 49(2):1–50, 2016b.

Paula Branco, Luís Torgo, and Rita P Ribeiro. Smogn: a pre-processing approach for imbalanced regression. In *First international workshop on learning with imbalanced domains: Theory and applications*, pp. 36–50. PMLR, 2017.

Paula Branco, Luis Torgo, and Rita P Ribeiro. Pre-processing approaches for imbalanced distributions in regression. *Neurocomputing*, 343:76–99, 2019.

M. Buda, A. Maki, and M. A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Advances in Knowledge Discovery and Data Mining: 13th Pacific-Asia Conference, PAKDD 2009 Bangkok, Thailand, April 27-30, 2009 Proceedings 13*, pp. 475–482. Springer, 2009.

Luís Camacho, Georgios Douzas, and Fernando Bacao. Geometric smote for regression. *Expert Systems with Applications*, pp. 116387, 2022.

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems*, 32, 2019.

Arthur Charpentier and Emmanuel Flachaire. Log-transform kernel density estimation of income distribution. *L'Actualité économique*, 91(1):141–159, 2015.

Arthur Charpentier and Abder Oulidi. Beta kernel quantile estimators of heavy-tailed loss distributions. *Statistics and computing*, 20(1):35–55, 2010.

Chawla, Bowyer, Hall, and Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

Song Xi Chen. Probability density function estimation using gamma kernels. *Annals of the Institute of Statistical Mathematics*, 52:471–480, 2000.

Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie. Classbalanced loss based on effective number of samples. *CVPR*, 2019.

Daniela De Angelis and G Alastair Young. Smoothing the bootstrap. *International Statistical Review/Revue Internationale de Statistique*, pp. 45–56, 1992.

Daniele De Martini and Fabio Rapallo. On multivariate smoothed bootstrap consistency. *Journal of statistical planning and inference*, 138(6):1828–1835, 2008.

Yifei Ding, Minping Jia, Jichao Zhuang, and Peng Ding. Deep imbalanced regression using cost-sensitive learning and deep feature transfer for bearing remaining useful life estimation. *Applied Soft Computing*, 127:109271, 2022.

Georgios Douzas, Fernando Bacao, and Felix Last. Improving imbalanced learning through a heuristic oversampling method based on k-means and smote. *Information Sciences*, 465:1–20, 2018.

Tarn Duong and Martin L Hazelton. Cross-validation bandwidth matrices for multivariate kernel density estimation. *Scandinavian Journal of Statistics*, 32(3):485–506, 2005.

Dina Elreedy, Amir F Atiya, and Firuz Kamalov. A theoretical distribution analysis of synthetic minority oversampling technique (smote) for imbalanced learning. *Machine Learning*, pp. 1–21, 2023.

M Falk and R-D Reiss. Weak convergence of smoothed and nonsmoothed bootstrap quantile estimates. *The Annals of Probability*, pp. 362–371, 1989.

Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C Prati, Bartosz Krawczyk, and Francisco Herrera. *Learning from imbalanced data sets*, volume 10. Springer, 2018a.

Alberto Fernández, Salvador Garcia, Francisco Herrera, and Nitesh V Chawla. Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of artificial intelligence research*, 61:863–905, 2018b.

Yu Gong, Greg Mori, and Frederick Tung. Ranksim: Ranking similarity regularization for deep imbalanced regression. *arXiv preprint arXiv:2205.15236*, 2022.

Xiaowei Gu, Plamen P Angelov, and Eduardo A Soares. A self-adaptive synthetic over-sampling technique for imbalanced classification. *International Journal of Intelligent Systems*, 35(6):923–943, 2020.

Peter Hall, Thomas J DiCiccio, and Joseph P Romano. On smoothing and the bootstrap. *The Annals of Statistics*, pp. 692–704, 1989.

Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *Advances in Intelligent Computing: International Conference on Intelligent Computing, ICIC 2005, Hefei, China, August 23-26, 2005, Proceedings, Part I 1*, pp. 878–887. Springer, 2005.

Tristen Hayfield and Jeffrey S Racine. Nonparametric econometrics: The np package. *Journal of statistical software*, 27:1–32, 2008.

Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pp. 1322–1328. IEEE, 2008.

C. Huang, Y. Li, C. Change Loy, and X. Tang. Learning deep representation for imbalanced classification. *CVPR*, 2016.

Bartosz Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.

Erin LeDell and Sebastien Poirier. H2O AutoML: Scalable automatic machine learning. *7th ICML Workshop on Automated Machine Learning (AutoML)*, July 2020. URL https://www.automl.org/wp-content/uploads/2020/07/AutoML_2020_paper_61.pdf.

Lee and Sauchi. Noisy replication in skewed binary classification. *Computational Statistics and Data Analysis*, 34(2):165–191, 2000.

Menardi and Torelli. Training and assessing classification rules with imbalanced data. *Data Mining and Knowledge Discovery*, 28(1):92–122, 2014.

Rita P Ribeiro and Nuno Moniz. Imbalanced regression and extreme value prediction. *Machine Learning*, 109:1803–1835, 2020.

David W Scott. *Multivariate density estimation: theory, practice, and visualization.* John Wiley & Sons, 2015.

Snigdha Sen, Krishna Pratap Singh, and Pavan Chakraborty. Dealing with imbalanced regression problem for large dataset using scalable artificial neural network. *New Astronomy*, 99:101959, 2023.

Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

BW Silverman and GA Young. The bootstrap: to smooth or not to smooth? *Biometrika*, 74(3):469–479, 1987.

Adrian FM Smith and Alan E Gelfand. Bayesian statistics without tears: a sampling–resampling perspective. *The American Statistician*, 46(2):84–88, 1992.

Sobom Matthieu Someé. *Estimations non paramétriques par noyaux associés multivariés et applications.* PhD thesis, Université de Franche-Comté, 2015.

Xin Yue Song, Nam Dao, and Paula Branco. Distsmogn: Distributed smogn for imbalanced regression problems. In *Fourth International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pp. 38–52. PMLR, 2022.

Michael Steininger, Konstantin Kobs, Padraig Davidson, Anna Krause, and Andreas Hotho. Density-based weighting for imbalanced regression. *Machine Learning*, 110:2187–2211, 2021.

Samuel Stocksieker, Denys Pommeret, and Arthur Charpentier. Data augmentation for imbalanced regression. In *International Conference on Artificial Intelligence and Statistics*, pp. 7774–7799. PMLR, 2023.

Bo Tang and Haibo He. Kerneladasyn: Kernel based adaptive synthetic data generation for imbalanced learning. In *2015 IEEE congress on evolutionary computation (CEC)*, pp. 664–671. IEEE, 2015.

Tomek. Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(11): 769–772, 1976. doi: 10.1109/TSMC.1976.4309452.

Luis Torgo and Rita Ribeiro. Utility-based regression. In *PKDD*, volume 7, pp. 597–604. Springer, 2007.

Luís Torgo, Rita P Ribeiro, Bernhard Pfahringer, and Paula Branco. Smote for regression. In *Portuguese conference on artificial intelligence*, pp. 378–389. Springer, 2013.

Ziyan Wang and Hao Wang. Variational imbalanced regression: Fair uncertainty quantification via probabilistic smoothing. *Advances in Neural Information Processing Systems*, 36, 2024.

Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-2(3):408–421, 1972. doi: 10.1109/TSMC.1972.4309137.

C. F. J. Wu. Jackknife, Bootstrap and Other Resampling Methods in Regression Analysis. *The Annals of Statistics*, 14(4):1261 – 1295, 1986. doi: 10.1214/aos/1176350142. URL `https://doi.org/10.1214/aos/1176350142`.

Wenglei Wu, Nicholas Kunz, and Paula Branco. Imbalancedlearningregression-a python package to tackle the imbalanced regression problem. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2022, Grenoble, France, September 19–23, 2022, Proceedings, Part VI*, pp. 645–648. Springer, 2023.

Y. Yang and Z. Xu. Rethinking the value of labels for improving class-imbalanced learning. *NeurIPS*, 2020.

Yuzhe Yang, Kaiwen Zha, Ying-Cong Chen, Hao Wang, and Dina Katabi. Delving into deep imbalanced regression, 2021.

Huaxiu Yao, Yiping Wang, Linjun Zhang, James Y Zou, and Chelsea Finn. C-mixup: Improving generalization in regression. *Advances in Neural Information Processing Systems*, 35:3361–3376, 2022.