ExNN-SMOTE: Extended Natural Neighbors Based SMOTE to Deal with Imbalanced Data

Hongjiao Guan* GUANHONGJIAO2008@163.COM
Bin Ma SDDXMB@126.COM
School of Cyber Security, Qilu University of Technology (Shandong Academy of Sciences), Shandong Computer Science Center (National Supercomputer Center in Jinan), Jinan 250353, China
Shandong Provincial Key Laboratory of Computer Networks, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250353, China

Yingtao Zhang YINGTAO@HIT.EDU.CN
Xianglong Tang TANGXL@HIT.EDU.CN
School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China

Editors: Vineeth N Balasubramanian and Ivor Tsang

Abstract

Many practical applications suffer from the problem of imbalanced classification. The minority class has poor classification performance; on the other hand, its misclassification cost is high. One reason for classification difficulty is the intrinsic complicated distribution characteristics (CDCs) in imbalanced data itself. Classical oversampling method SMOTE generates synthetic minority class examples between neighbors, which is parameter dependent. Furthermore, due to blindness of neighbor selection, SMOTE suffers from overgeneralization in the minority class. To solve such problems, we propose an oversampling method, called extended natural neighbors based SMOTE (ExNN-SMOTE). In ExNN-SMOTE, neighbors are determined adaptively by capturing data distribution characteristics. Extensive experiments over synthetic and real datasets demonstrate the effectiveness of ExNN-SMOTE dealing with CDCs and the superiority of ExNN-SMOTE over other SMOTE-related methods.

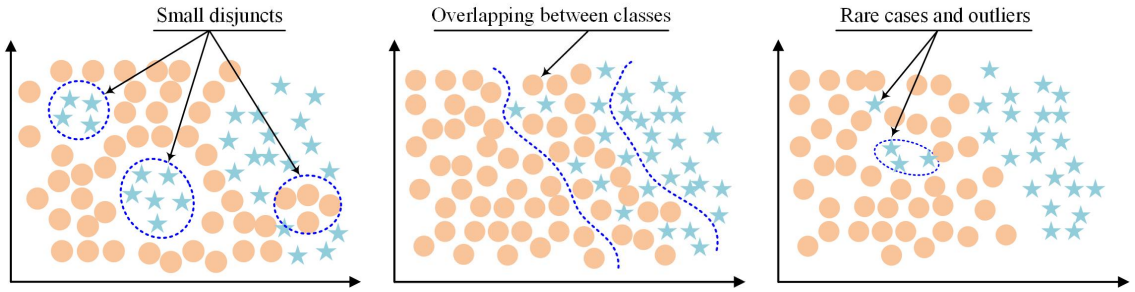
Keywords: Imbalanced classification, SMOTE, Extended natural neighbors, Complicated distribution characteristics, Decision tree

1. Introduction

Imbalanced classification has always been a challenging issue in data mining and machine learning. It exists commonly in practical applications, such as fault diagnosis, credit assessment, and so on Tian et al. (2021); Wang et al. (2021). In two-class classification problems, imbalance means that the number of examples in one class (often called negative or majority class) is far more than that of examples in the other class (usually called positive or minority class). Traditional classification methods have deteriorative performance when dealing with imbalanced datasets Du et al. (2020). In particular, the minority class has a seriously low recognition rate. However, the misclassification cost of the positive class is usually higher than that of the negative class, and the minority class is the more focused category in practice. This brings the contradiction between the requirement of a high recognition rate and the difficulty of correct classification.

There are two aspects of reasons: algorithmic-level and data-level. Traditional statistical learning methods assume a roughly equal number of examples and the same misclassification cost in each class. The goal of these methods is to obtain a low empirical risk, i.e., a low error rate on training data. Therefore, when dealing with imbalanced data, the classification models favor the majority class, leading to poor generalization performance of the minority class. Ensemble learning and cost-sensitive learning are commonly used to deal with imbalanced classification. A major drawback of cost-sensitive learning is that the knowledge of costs for each class is usually unavailable in practice. Resampling-based ensemble methods have recently received lots of attention, such as DPHS-MDS Gao et al. (2020) and NUS-SE Seng et al. (2021).

Another factor for the imbalanced classification problem is complicated distribution characteristics (CDCs) in imbalanced data itself Napierala and Stefanowski (2016). These CDCs mainly include small disjuncts Holte et al. (1989), overlapping between classes Prati et al. (2004), rare cases and outliers in the minority class Napierala and Stefanowski (2016), which are shown in Fig. 1. Small disjuncts mean small clusters of a different label in the homogeneous region. Usually, small disjuncts of the minority class occur in the majority class due to the sparsity of the minority class. Small disjuncts are likely to cause overfitting or data fragmentation when using decision trees to classify, for instance. Overlapping between classes indicates that the examples from different classes have similar features. Following the global imbalance, the overlapping area has imbalanced class distribution, leading to the bias toward the majority class. Rare cases are isolated pairs or triples of minority examples, which are distant from the decision border. Outliers are single minority examples, which represent rare but valuable sub-concepts. Rare cases and outliers are the result of the underrepresented minority class. These examples are easy to be considered as noisy ones, thus suffering from low generalization performance.



(a) Small disjuncts (b) Overlapping between classes (c) Rare cases and outliers

Figure 1: Illustrations of complex distribution characteristics in imbalanced data. (Circles and stars indicate the majority and minority class examples, respectively)

Generally speaking, these CDCs also exist in balanced datasets. However, their negative influence on classification performance is aggravated by class imbalance and the scarcity of positive examples in imbalanced data He and Garcia (2008). Furthermore, these CDCs do not occur alone in real datasets. Therefore, traditional algorithms have great difficulty in dealing with imbalanced datasets. Many resampling methods have been proposed, including undersampling, oversampling, and hybrid sampling approaches. Random undersampling

and random oversampling are the simplest, which deletes majority class examples or copy minority class examples randomly. Random resampling leads to the loss of important information or overfitting, thus obtaining unsatisfactory classification performance. Among oversampling methods, the classical synthetic minority oversampling technique (SMOTE) Chawla et al. (2002) and its extensions Han et al. (2005); Bunkhumpornpat et al. (2009); Maciejewski and Stefanowski (2011) are popular ones, which take into account the local neighborhoods of the minority and majority class examples to generate synthetic instances.

These SMOTE-based methods generate synthetic instances using linear interpolation between each seed example and its k-nearest neighbors. Therefore, these methods are parameter-dependent. Appropriate parameter k differs among different datasets. It is inefficient to find the suitable parameter for each dataset. Recently, Li et al. Li et al. (2021) proposed a SMOTE method based on natural neighbors (NaNSMOTE). In NaNSMOTE, synthetic minority instances are generated between the seed example and its natural neighbors. However, due to the sparsity of the minority class, NaNSMOTE is deficient in representing the distribution of the minority class using natural neighbors.

In this paper, an extended natural neighbors based SMOTE (ExNN-SMOTE) is proposed to oversample the imbalanced datasets. In ExNN-SMOTE, synthetic examples are generated between the seed minority instances and their extended natural neighbors (ExNNs). The ExNNs of an example are determined adaptively using λ-nearest neighbors and reverse λ-nearest neighbors, where λ is the natural neighbor eigenvalue. In addition, we provide an analysis of how the proposed ExNN-SMOTE is beneficial for the improvement of imbalanced classification. Extensive experiments over synthetic and real datasets demonstrate the effectiveness of ExNN-SMOTE compared with the traditional SMOTE and the latest NaNSMOTE.

The rest of this paper is organized as follows. Section 2 gives the preliminary knowledge about natural neighbors. Section 3 presents the details of the proposed ExNN-SMOTE method. Section 4 reports and analyzes the results of extensive comparative experiments. Finally, the conclusions are presented in Section 5.

2. Preliminaries

Since the methods in this paper are associated with the natural neighbor (NaN), we introduce its concept and list some related notations.

2.1. Notations

Assume there are n examples, including n_{pos} minority class instances and n_{neg} majority class instances.

- $Tr = Pos \cup Neg = \{x_1, x_2, \ldots, x_{n_{pos}}\} \cup \{x_{n_{pos}+1}, \ldots, x_n\}$. The training set Tr contains the minority class Pos and the majority class Neg. $x_i (i = 1, 2, \ldots, n)$ is the ith example with d features.
- $NN_r(x_i) = \{x_j, \ldots\}$ denotes the set of r-nearest neighbors of x_i.

- $RNN_r(x_i) = \{x_k, \dots\}$ indicates the set of reverse r-nearest neighbors of x_i. If example x_i is one of r-nearest neighbors of example x_k, x_k is one of reverse r-nearest neighbors of x_i; i.e., $x_i \in NN_r(x_k) \iff x_k \in RNN_r(x_i)$.
- $nb(x_i) = t_i$ records the times that example x_i is the neighbor of other examples, i.e., the number of examples in $RNN_r(x_i)$. The number of each example's reverse nearest neighbors (the size of $RNN_r(x_i)$) is not necessarily r. By contrast, the size of $NN_r(x_i)$ must be r.
- $NaN(x_i) = \{x_l, \dots\}$ is the set of natural neighbors of x_i.
- $ExNN(x_i) = \{x_m, \dots\}$ is the set of extended natural neighbors of x_i.

Considering that nominal and continuous attributes may exist simultaneously, the heterogeneous value difference metric (HVDM) is used to calculate the distances and obtain the neighborhood. Refer to Guan et al. (2021) for more details.

2.2. Natural Neighbors

The natural neighbor (NaN) Zhu et al. (2016) is a novel concept, which is first proposed in 2016. Different from nearest neighbor (NN) and reverse nearest neighbor (RNN), NaN is parameter-independent and it can reflect data distribution characteristics. Due to the merits of NaN, it has been widely employed in machine learning tasks, such as classification Li et al. (2021, 2019), clustering Cheng et al. (2019, 2017), instance reduction Yang et al. (2018); Zhao and Li (2020), outlier detection Huang et al. (2017), and so on.

NaN is inspired by friendship in human society. When two persons are friends of each other, they are true friends. If every person (except strangers) has at least one true friend, the natural stable structure (NSS) is established Zhu et al. (2016). Analogy to human society, if example x is one of λ nearest neighbors of example y and y is one of λ nearest neighbors of x, x (y) is called a natural neighbor of y (x). When each example (except noises) has at least one NaN, NSS is obtained. According to Definition 1, $NaN(x)$ is the intersection of $NN_\lambda(x)$ and $RNN_\lambda(x)$.

Definition 1 *(Natural Neighbor)*
$y \in NaN(x) \iff y \in NN_\lambda(x) \wedge x \in NN_\lambda(y) \iff y \in NN_\lambda(x) \wedge y \in RNN_\lambda(x)$

In Definition 1, λ is called as natural neighbor eigenvalue (NaNE), which is the minimal search number r to establish the NSS. That is, when the neighbor number r increases from 1 to λ, all examples (except noises) have their NaNs. Therefore, the value of λ is associated with data distribution, and thus it varies from datasets. Large λ indicates relatively sparse sample space. The pseudocode of searching NaNs is shown in Algorithm 1.

Definition 2 *(Natural Neighbor Eigenvalue)*
$\lambda = argmin_r\{(\forall r \in N)(\forall x)(\exists y \neq x), (x \in NN_r(y)) \wedge (y \in NN_r(x))\}$

Algorithm 1: Searching natural neighbors (NaN_Search)
Input: Tr, training set.
Output: NaN, natural neighbors of each example.

Initialization: $r = 1, \forall x_i \in Tr, NN_r(x_i) = \varnothing, RNN_r(x_i) = \varnothing, nb(x_i) = 0$;
while *True* **do**
  foreach $x_i \in Tr$ **do**
    Find r-th nearest neighbor x_j of x_i in Tr;
    $NN_r(x_i) = NN_r(x_i) \cup \{x_j\}, RNN_r(x_j) = RNN_r(x_j) \cup \{x_i\}, nb(x_j) = nb(x_j) + 1$;
  end
  Compute the number n of examples with $nb(x_i) == 0$;
  if n does not change **then**
    $\lambda = r$;
    foreach $x_i \in Tr$ **do**
      $NaN(x_i) = NN_\lambda(x_i) \cap RNN_\lambda(x_i)$;
    end
    return NaN;
  else
    $r = r + 1$;
  end
end

3. Proposed Method

3.1. Extended Natural Neighbors

According to Algorithm 1, the nature of example x's NaNs is the intersection of its nearest neighbors and reverse nearest neighbors. For imbalanced data, the minority class examples are inherently sparse. When searching NaNs in the minority class space, the NaN set of a positive example may be empty or contains only a few instances. If a NaN set is empty, the example is considered as noise, and no synthetic instances are generated. It is claimed that the minority class examples represent subconcepts that are underrepresented in the original imbalanced data, so each minority class example should be treated carefully Guan et al. (2021). If NaNs are adopted as neighbors in SMOTE, the synthetic examples are likely to surround the seed example, forming small clusters. Hence, NaN is a tough neighborhood.

We propose the extended natural neighbor (ExNN) to loosen NaN's neighborhood. If example x is one of λ-nearest neighbors of example y or y is one of λ-nearest neighbors of x, x (y) is called an extended natural neighbor of y (x). According to Definition 3, $ExNN(x)$ is the union of $NN_\lambda(x)$ and $RNN_\lambda(x)$. Hence, $NaN(x) \subseteq ExNN(x)$. ExNN can analogize knowledge relationship. If one person x is a friend of another person y but y is not a friend of x, x and y are knowledge relationship. There are similarities between people who know each other. ExNN is a looser relationship than NaN. Note that the definition of λ in ExNN is the same as that in NaN. Searching ExNNs is similar to searching NaNs and the pseudocode is shown in Algorithm 2. The difference is that ExNNs are the union set of nearest neighbors and reverse nearest neighbors.

Definition 3 *(Extended Natural Neighbor)*
$y \in ExNN(x) \iff y \in NN_\lambda(x) \vee x \in NN_\lambda(y) \iff y \in NN_\lambda(x) \vee y \in RNN_\lambda(x)$

Algorithm 2: Searching extended natural neighbors (ExNN_Search)
Input: Tr, training set.
Output: $ExNN$, extended natural neighbors of each example.

Initialization: $r = 1, \forall x_i \in Tr, NN_r(x_i) = \varnothing, RNN_r(x_i) = \varnothing, nb(x_i) = 0$;
while *True* **do**
  foreach $x_i \in Tr$ **do**
    Find r-th nearest neighbor x_j of x_i in Tr;
    $NN_r(x_i) = NN_r(x_i) \cup \{x_j\}, RNN_r(x_j) = RNN_r(x_j) \cup \{x_i\}, nb(x_j) = nb(x_j) + 1$;
  end
  Compute the number n of examples with $nb(x_i) == 0$;
  if n does not change **then**
    $\lambda = r$;
    foreach $x_i \in Tr$ **do**
      $ExNN(x_i) = NN_\lambda(x_i) \cup RNN_\lambda(x_i)$;
    end
    return $ExNN$;
  else
    $r = r + 1$;
  end
end

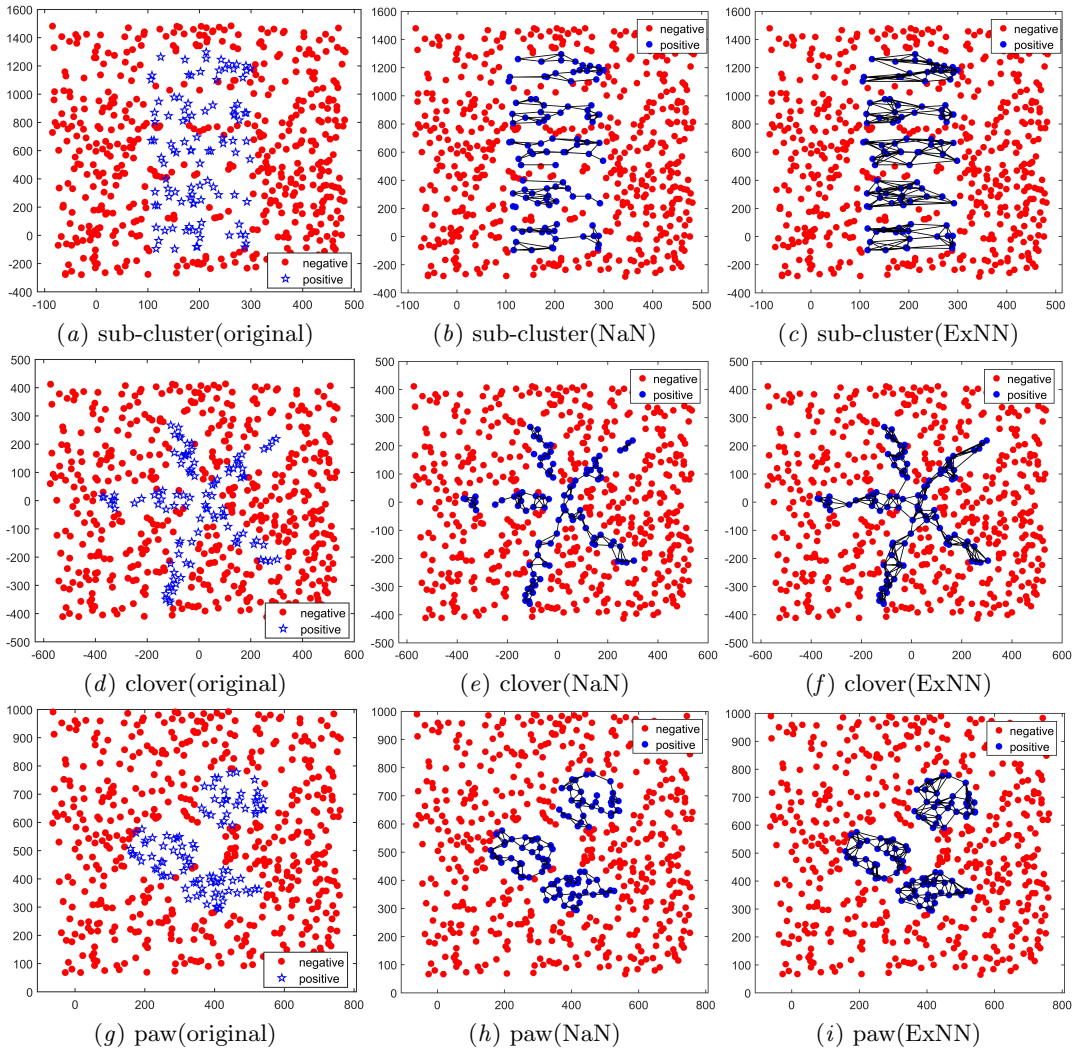
Figure 2 shows the NaNs and ExNNs of positive instances in three synthetic datasets (sub-cluster, clover, and paw). These three datasets contain small disjuncts of different shapes. The description of the three synthetic datasets is provided in Section 4.1. In Fig. 2, the left column lists original sample distribution; the middle and right columns show NaN and ExNN neighborhoods, respectively. If two examples are neighbors, they are connected using a black solid line. From Fig. 2, we can see that ExNN explores broader local structures than NaN.

3.2. ExNN-SMOTE

SMOTE generates synthetic examples using linear interpolation between the positive seed instance and its k neighbors of the same class. Due to parameter-dependence of k and blindness of selecting neighbors, SMOTE suffers from overgeneralization for the minority class examples Maciejewski and Stefanowski (2011); Sáez et al. (2015). That is, the synthetic minority class examples generated by traditional SMOTE may occur in the space of the majority class, leading to overlapping between classes and blurring the class boundaries. This issue ascribes that the neighbors in SMOTE cannot reflect the local structure correctly.

NaNSMOTE Li et al. (2021) is a parameter-free SMOTE method, which uses NaNs as neighbors. It relieves the problem of overgeneralization, since natural neighbors capture the local distribution. However, as mentioned in Section 3.1, NaNSMOTE is likely to generate small clusters surrounding the positive seed examples. ExNN-SMOTE is proposed to deal with this problem. ExNN-SMOTE is to generate new examples between the seed and its ExNNs. The pseudocode of ExNN-SMOTE is presented in Algorithm 3.

Now we explain the superiority of ExNN-SMOTE over NaNSMOTE. From Figure 2, we can observe that ExNN explores broader local structures than NaN. That is to say, ExNN captures mores neighbors while it maintains local structures. Hence, ExNN-SMOTE



(a) sub-cluster(original) (b) sub-cluster(NaN) (c) sub-cluster(ExNN)

(d) clover(original) (e) clover(NaN) (f) clover(ExNN)

(g) paw(original) (h) paw(NaN) (i) paw(ExNN)

Figure 2: Distributions of NaNs and ExNNs of positive instances in three synthetic datasets.

is inclined to generate more effective borderline positive examples than NaNSMOTE (This will be validated in Section 4.2). There are two kinds of borderline examples Sáez et al. (2015): examples in overlapping areas and examples very close to the difficult shape of the boundary. Overgeneralization means it generates the former and effective borderline examples indicate the latter. On the one hand, due to local structure preservation, the problem of overgeneralization is relieved in ExNN-SMOTE. On the other hand, when the same number of positive examples needs to be generated, more effective borderline examples can obtain more clear class boundaries.

Algorithm 3: Using ExNN-SMOTE to preprocess imbalanced data
Input: Tr, training set.
Output: New_Tr, new training set after using ExNN-SMOTE.

Divide Tr into positive and negative subsets: $Tr = Pos \cup Neg$;
Compute the number n_{new} of minority class examples needed to be generated: $n_{new} = n_{neg} - n_{pos}$;
For each $x_i \in Pos$, find its ExNNs in the minority class: $ExNN(x_i) = \text{ExNN\_Search}(Pos)$;
Generate synthetic minority class examples between each original positive instance and its ExNNs using linear interpolation: $New\_Pos = \text{SMOTE}(Pos, ExNN)$;
$New\_Tr = New\_Pos \cup Neg$.

4. Experimental Results and Analysis

Experiments over synthetic and real datasets are implemented to validate the effectiveness of ExNN-SMOTE and to compare ExNN-SMOTE with other SMOTE-related methods.

4.1. Experimental Datasets and Setups

Three groups of synthetic datasets and ten real-world datasets are used in the experiment. The synthetic datasets derive from three datasets with different shapes of small disjuncts: sub-cluster, clover, and paw Stefanowski (2013); Napierała et al. (2010). In each group, ten datasets are configured with two factors: imbalance ratio (IR) and disturbance ratio (DR). Imbalance ratio is used to measure the imbalance degree, which is defined as the sample number of the majority class divided by that of the minority class. Disturbance ratio indicates the degree of overlapping between classes or disturbing the borders of sub-regions in the minority class. Imbalanced datasets with two IRs are considered: datasets with $IR = 5$ have 600 examples and datasets with $IR = 7$ have 800 examples. Five levels of DR are considered: 0%, 30%, 50%, 60%, and 70%, respectively. All the synthetic imbalanced datasets are two dimensions in the numerical space. In Fig. 2, synthetic datasets with $IR = 5$, $DR = 0\%$ have been shown in the left column.

Ten real-world datasets are available from the UCI repository. Their characteristics are shown in Table 1. To understand the classification difficulty of imbalanced datasets explicitly, according to the method proposed by Napierala Napierala and Stefanowski (2016), all positive (negative) examples are divided into four types: safe, borderline, rare, and outlying examples. Tables 2 and 3 list the numbers and proportions of the four type examples in the majority and minority classes, respectively. From the two tables, it can be observed that most majority class examples are safe. For the minority class, most instances are unsafe (including borderline, rare, outlying examples). Especially, almost all positive examples are unsafe in dataset *cleveland*, *glass*, *solar-flare*, and *yeast*, which are difficult to classify.

In this paper, ExNN-SMOTE is compared with traditional SMOTE ($k = 3, 5, 7$) and the latest NaNSMOTE. Five runs of a stratified five-fold cross-validation are performed and the averaged results are obtained. The decision tree algorithm CART without pruning is used to construct the classifier. The minimum number of examples per leaf node is set as 2. The classification performance is evaluated using the area under the ROC curve (AUC) and

Table 1: Characteristics of 10 real datasets.

Dataset	Instances	Minority	IR	Attributes
satimage	6435	626	9.3	36
credit-g	1000	300	2.33	20
ecoli	336	35	8.6	7
cmc	1473	333	3.4	9
cleveland	303	35	7.66	13
glass	214	17	11.59	9
abalone	4177	335	11.47	8
solar-flare	1066	43	23.79	12
transfusion	748	178	3.2	4
yeast	1484	51	28.1	8

Table 2: Numbers and ratios of four types of majority class examples.

	Safe		Borderline		Rare		Outlying	
Dataset	No.	Ratio(%)	No.	Ratio(%)	No.	Ratio(%)	No.	Ratio(%)
satimage	3782	**94.08**	196	4.88	34	0.85	8	0.20
credit-g	466	**66.57**	204	29.14	26	3.71	4	0.57
ecoli	280	**93.02**	16	5.32	3	1.00	2	0.66
cmc	819	**71.84**	283	24.82	28	2.46	10	0.88
cleveland	235	**89.69**	26	9.92	1	0.38	0	0.00
glass	184	**93.40**	12	6.09	1	0.51	0	0.00
abalone	3636	**94.64**	182	4.74	20	0.52	4	0.10
solar-flare	991	**96.87**	31	3.03	1	0.10	0	0.00
transfusion	442	**77.54**	107	18.77	17	2.98	4	0.70
yeast	1412	**98.53**	18	1.26	3	0.21	0	0.00

Table 3: Numbers and ratios of four types of minority class examples.

	Safe		Borderline		Rare		Outlying	
Dataset	No.	Ratio(%)	No.	Ratio(%)	No.	Ratio(%)	No.	Ratio(%)
satimage	197	**47.47**	135	32.53	49	11.81	34	8.19
credit-g	47	15.67	150	**50.00**	71	23.67	32	10.67
ecoli	9	25.71	16	**45.71**	7	20.00	3	8.57
cmc	46	13.81	142	**42.64**	83	24.92	62	18.62
cleveland	0	0.00	12	34.29	7	20.00	16	**45.71**
glass	0	0.00	4	23.53	8	**47.06**	5	29.41
abalone	39	11.64	119	**35.52**	77	22.99	100	29.85
solar-flare	1	2.33	12	27.91	13	30.23	17	**39.53**
transfusion	33	18.54	64	**35.96**	40	22.47	41	23.03
yeast	2	3.92	19	37.25	10	19.61	20	**39.22**

sensitivity. AUC is the arithmetic mean of sensitivity and specificity López et al. (2013). All the experiments are performed using Matlab R2017a.

Table 4: AUC results of decision tree on synthetic datasets.

Dataset	SMOTE3	SMOTE5	SMOTE7	NaNSMOTE	ExNN-SMOTE
sub-clus$_{IR=5,DR=0}$	0.9538	0.9432	0.9434	**0.9608**	0.9416
sub-clus$_{IR=5,DR=30}$	0.7860	0.8046	0.7936	0.8066	**0.8072**
sub-clus$_{IR=5,DR=50}$	0.7372	**0.7526**	0.7436	0.7266	0.7504
sub-clus$_{IR=5,DR=60}$	0.7484	0.7422	0.7576	0.7500	**0.7578**
sub-clus$_{IR=5,DR=70}$	0.7344	0.7186	**0.7540**	0.7306	0.7472
sub-clus$_{IR=7,DR=0}$	**0.9634**	0.9561	0.9436	0.9581	0.9443
sub-clus$_{IR=7,DR=30}$	0.8099	0.8069	0.8076	0.8006	**0.8194**
sub-clus$_{IR=7,DR=50}$	0.7273	0.7391	**0.7543**	0.7333	0.7494
sub-clus$_{IR=7,DR=60}$	0.7143	0.7289	0.7384	**0.7441**	0.7231
sub-clus$_{IR=7,DR=70}$	0.6894	0.7087	0.7234	0.7050	**0.7286**
clover$_{IR=5,DR=0}$	0.8666	0.8690	0.8722	0.8640	**0.8914**
clover$_{IR=5,DR=30}$	0.8058	0.7984	**0.8156**	0.7856	0.8032
clover$_{IR=5,DR=50}$	0.7540	0.7582	**0.7658**	0.7380	0.7558
clover$_{IR=5,DR=60}$	0.7600	0.7604	**0.7856**	0.7580	0.7710
clover$_{IR=5,DR=70}$	0.7406	0.7536	0.7330	0.7362	**0.7568**
clover$_{IR=7,DR=0}$	0.8711	**0.8924**	0.8876	0.8810	0.8774
clover$_{IR=7,DR=30}$	0.8040	**0.8151**	0.8127	0.7963	0.8059
clover$_{IR=7,DR=50}$	0.7440	0.7584	0.7583	0.7431	**0.7639**
clover$_{IR=7,DR=60}$	0.7219	0.7107	**0.7551**	0.7544	0.7503
clover$_{IR=7,DR=70}$	0.7321	0.7266	0.7443	0.7201	**0.7451**
paw$_{IR=5,DR=0}$	0.9372	0.9274	0.9428	**0.9468**	0.9336
paw$_{IR=5,DR=30}$	0.8344	**0.8456**	0.8328	0.8202	0.8380
paw$_{IR=5,DR=50}$	0.8098	0.8138	0.8100	0.8070	**0.8168**
paw$_{IR=5,DR=60}$	0.7726	0.7802	0.7788	0.7704	**0.7952**
paw$_{IR=5,DR=70}$	0.7838	0.7888	0.7914	0.7722	**0.7916**
paw$_{IR=7,DR=0}$	0.9356	**0.9376**	0.9299	0.9266	0.9329
paw$_{IR=7,DR=30}$	0.8391	0.8440	0.8449	0.8241	**0.8469**
paw$_{IR=7,DR=50}$	0.7781	**0.8046**	0.8001	0.7926	0.7894
paw$_{IR=7,DR=60}$	0.7601	0.7759	0.7799	0.7520	**0.7860**
paw$_{IR=7,DR=70}$	0.7586	0.7744	**0.7861**	0.7576	0.7651
Average value	0.7958	0.8012	**0.8062**	0.7954	**0.8062**
Average rank	3.67	2.80	2.37	3.97	**2.20**
Adjusted p-value	9.82E-04*	0.2833	0.6831	6.03E-05*	–

4.2. Results on Synthetic Datasets

Tables 4 and 5 list the results of AUC and sensitivity, respectively, when using SMOTE ($k = 3, 5, 7$), NaNSMOTE, and ExNN-SMOTE to preprocess synthetic datasets. The best case for each dataset is highlighted in bold. The average values, average ranks, and adjusted p-values obtained by the Friedman test are also presented. The smaller the average rank, the better the method. It can be observed that: (1) For AUC, ExNN-SMOTE has the smallest rank. According to the Friedman test ($p < 0.05$), ExNN-SMOTE is significantly better than SMOTE ($k = 3$) and NaNSMOTE. When $k = 5$ and $k = 7$, SMOTE achieves comparative AUC results compared with ExNN-SMOTE. (2) SMOTE with $k = 7$ performs best in terms of sensitivity. ExNN-SMOTE has the second lowest rank and ExNN-SMOTE does not show a significant difference compared with SMOTE ($k = 7$).

From Tables 4 and 5, we also can observe that NaNSMOTE has the largest ranks in terms of AUC and sensitivity when using decision trees for classification. To further explain the

Table 5: Sensitivity results of decision tree on synthetic datasets.

Dataset	SMOTE3	SMOTE5	SMOTE7	NaNSMOTE	ExNN-SMOTE
sub-clus$_{IR=5,DR=0}$	0.9420	0.9280	**0.9440**	0.9360	0.9040
sub-clus$_{IR=5,DR=30}$	0.6920	**0.7480**	0.7340	0.7280	0.7400
sub-clus$_{IR=5,DR=50}$	0.6220	0.6660	0.6540	0.6200	**0.6700**
sub-clus$_{IR=5,DR=60}$	0.6560	0.6540	**0.6920**	0.6560	0.6780
sub-clus$_{IR=5,DR=70}$	0.6160	0.6220	**0.6900**	0.6340	0.6720
sub-clus$_{IR=7,DR=0}$	0.9620	**0.9480**	0.9440	0.9240	0.9020
sub-clus$_{IR=7,DR=30}$	0.7320	0.7360	0.7560	0.7080	**0.7600**
sub-clus$_{IR=7,DR=50}$	0.5940	0.6280	**0.6660**	0.6120	0.6540
sub-clus$_{IR=7,DR=60}$	0.5720	0.6200	**0.6420**	0.6360	0.6060
sub-clus$_{IR=7,DR=70}$	0.5420	0.5940	**0.6320**	0.5740	0.6280
clover$_{IR=5,DR=0}$	0.8020	0.8060	0.8200	0.7960	**0.8520**
clover$_{IR=5,DR=30}$	0.7280	0.7160	**0.7560**	0.6980	0.7260
clover$_{IR=5,DR=50}$	0.6380	0.6640	**0.6880**	0.6180	0.6640
clover$_{IR=5,DR=60}$	0.6660	0.6820	**0.7260**	0.6680	0.7040
clover$_{IR=5,DR=70}$	0.6320	0.6700	0.6360	0.6200	**0.6860**
clover$_{IR=7,DR=0}$	0.7920	**0.8440**	0.8340	0.8160	0.8200
clover$_{IR=7,DR=30}$	0.7060	**0.7360**	0.7320	0.6860	0.7160
clover$_{IR=7,DR=50}$	0.6120	0.6480	0.6600	0.6240	**0.6720**
clover$_{IR=7,DR=60}$	0.5860	0.5800	**0.6640**	0.6520	0.6620
clover$_{IR=7,DR=70}$	0.6000	0.6000	**0.6460**	0.5860	0.6420
paw$_{IR=5,DR=0}$	0.9020	0.8820	0.9160	**0.9220**	0.8940
paw$_{IR=5,DR=30}$	0.7580	**0.7820**	0.7560	0.7220	0.7800
paw$_{IR=5,DR=50}$	0.7340	0.7440	0.7340	0.7280	**0.7580**
paw$_{IR=5,DR=60}$	0.6720	0.7080	0.7020	0.6720	**0.7320**
paw$_{IR=5,DR=70}$	0.6940	0.7180	**0.7300**	0.6800	0.7160
paw$_{IR=7,DR=0}$	0.8960	**0.8980**	0.8820	0.8760	0.8880
paw$_{IR=7,DR=30}$	0.7560	0.7720	0.7780	0.7160	**0.7900**
paw$_{IR=7,DR=50}$	0.6600	**0.7280**	0.7160	0.6840	0.7040
paw$_{IR=7,DR=60}$	0.6400	0.6780	0.6980	0.6240	**0.7060**
paw$_{IR=7,DR=70}$	0.6480	0.6880	**0.7200**	0.6500	0.6680
Average value	0.7017	0.7229	**0.7383**	0.7022	0.7331
Average rank	3.93	2.67	**1.92**	4.20	2.28
Adjusted p-value	2.35E-06*	0.1324	–	8.93E-08*	0.3691

superiority of ExNN-SMOTE over NaNSMOTE, the average numbers of NaNs and ExNNs in each synthetic dataset are counted, which is shown in Table 6. It can be seen that the neighbor number of each example is close to four in NaNSMOTE, whereas each example has an average of seven neighbors in ExNN-SMOTE. This partly demonstrates the superiority of ExNN-SMOTE because SMOTE with $k = 7$ performs better than SMOTE with $k = 3$ and $k = 5$ (Tables 4 and 5). Considering that the synthetic datasets are configured with different small disjuncts and several disturbance ratios, we conclude that ExNN-SMOTE can deal with CDCs better than other compared methods.

We calculates the ratios of four types of samples after using NaNSMOTE (M1) and ExNN-SMOTE (M2). Table 7 presents the results of the minority class of synthetic datasets with $DR = 60\%$. We can find that using ExNN-SMOTE obtains more borderline examples and less safe examples than using NaNSMOTE. This is consistent with the explanation in

Table 6: Average numbers of NaNs in NaNSMOTE (M1) and ExNNs in ExNN-SMOTE (M2) on synthetic datasets.

Dataset	M1	M2	Dataset	M1	M2	Dataset	M1	M2
$\text{sub-clus}_{\text{IR=5,DR=0}}$	3.97	6.96	$\text{clover}_{\text{IR=5,DR=0}}$	4.06	6.26	$\text{paw}_{\text{IR=5,DR=0}}$	3.78	5.36
$\text{sub-clus}_{\text{IR=5,DR=30}}$	3.78	6.28	$\text{clover}_{\text{IR=5,DR=30}}$	4.24	6.74	$\text{paw}_{\text{IR=5,DR=30}}$	3.36	5.62
$\text{sub-clus}_{\text{IR=5,DR=50}}$	3.94	6.05	$\text{clover}_{\text{IR=5,DR=50}}$	4.50	6.65	$\text{paw}_{\text{IR=5,DR=50}}$	3.77	6.10
$\text{sub-clus}_{\text{IR=5,DR=60}}$	3.49	5.78	$\text{clover}_{\text{IR=5,DR=60}}$	4.25	6.09	$\text{paw}_{\text{IR=5,DR=60}}$	3.54	5.33
$\text{sub-clus}_{\text{IR=5,DR=70}}$	3.84	5.95	$\text{clover}_{\text{IR=5,DR=70}}$	4.10	6.14	$\text{paw}_{\text{IR=5,DR=70}}$	3.52	5.23
$\text{sub-clus}_{\text{IR=7,DR=0}}$	3.98	6.80	$\text{clover}_{\text{IR=7,DR=0}}$	4.00	6.32	$\text{paw}_{\text{IR=7,DR=0}}$	3.55	5.49
$\text{sub-clus}_{\text{IR=7,DR=30}}$	3.90	6.48	$\text{clover}_{\text{IR=7,DR=30}}$	4.19	6.56	$\text{paw}_{\text{IR=7,DR=30}}$	3.23	5.73
$\text{sub-clus}_{\text{IR=7,DR=50}}$	3.72	6.41	$\text{clover}_{\text{IR=7,DR=50}}$	4.20	6.87	$\text{paw}_{\text{IR=7,DR=50}}$	3.91	6.00
$\text{sub-clus}_{\text{IR=7,DR=60}}$	3.57	5.52	$\text{clover}_{\text{IR=7,DR=60}}$	4.15	6.12	$\text{paw}_{\text{IR=7,DR=60}}$	3.49	5.62
$\text{sub-clus}_{\text{IR=7,DR=70}}$	3.82	5.81	$\text{clover}_{\text{IR=7,DR=70}}$	4.13	5.97	$\text{paw}_{\text{IR=7,DR=70}}$	3.58	5.26
Average	3.80	6.21						

Table 7: Numbers and proportions of four type examples in the minority class of synthetic datasets with $DR = 60\%$ (M1: NaNSMOTE, M2: ExNN-SMOTE)

		Safe		Borderline		Rare		Outlying	
Data	Method	No.	Ratio(%)	No.	Ratio(%)	No.	Ratio(%)	No.	Ratio(%)
1	None	14	14.00	44	44.00	31	31.00	11	11.00
	M1	**423**	**84.60**	74	14.80	3	0.60	0	0.00
	M2	385	77.00	**110**	**22.00**	4	0.80	1	0.20
2	None	13	13.00	35	35.00	34	34.00	18	18.00
	M1	**611**	**87.29**	80	11.43	9	1.29	0	0.00
	M2	556	79.43	**129**	**18.43**	13	1.86	2	0.29
3	None	11	11.00	53	53.00	25	25.00	11	11.00
	M1	**436**	**87.90**	53	10.69	6	1.21	1	0.20
	M2	422	84.40	**71**	**14.20**	5	1.00	2	0.40
4	None	5	5.00	49	49.00	29	29.00	17	17.00
	M1	**618**	**89.05**	72	10.37	2	0.29	2	0.29
	M2	564	80.57	**123**	**17.57**	13	1.86	0	0.00
5	None	37	37.00	32	32.00	20	20.00	11	11.00
	M1	**452**	**90.40**	40	8.00	7	1.40	1	0.20
	M2	439	87.80	**55**	**11.00**	5	1.00	1	0.20
6	None	24	24.00	34	34.00	28	28.00	14	14.00
	M1	**635**	**90.71**	61	8.71	3	0.43	1	0.14
	M2	604	86.29	**92**	**13.14**	3	0.43	1	0.14

Section 3.2. Borderline examples are more important than safe examples when determining decision borderlines.

4.3. Results on Real Datasets

Tables 8 and 9 are the results of AUC and sensitivity obtained by decision tree on the real datasets. The best case is highlighted in bold. The average values, average ranks,

and adjusted p-values obtained by the Friedman test are also presented. It is observed that ExNN-SMOTE achieves the best results and lowest average rank in terms of AUC and sensitivity. The star in the last row indicates that there is a statistical significance ($p < 0.05$) among the compared methods. The results of ExNN-SMOTE are significantly better than those of SMOTE and NaNSMOTE.

Table 8: AUC results of decision tree on real datasets.

Dataset	SMOTE3	SMOTE5	SMOTE7	NaNSMOTE	ExNN-SMOTE
satimage	0.7684	0.7668	0.7786	0.7717	**0.7871**
credit-g	0.6313	0.6353	0.6270	0.6363	**0.6393**
ecoli	0.7716	0.7714	0.7781	0.7660	**0.7901**
cmc	0.6019	0.6003	0.5978	0.6070	**0.6089**
cleveland	0.5622	0.5859	0.5873	**0.6302**	0.5902
glass	0.6974	0.6675	0.6789	0.6131	**0.7062**
abalone	0.6926	0.6864	0.6933	0.7038	**0.7089**
solar-flare	0.6340	**0.6521**	0.6373	0.6379	0.6490
transfusion	0.5996	0.6080	0.6098	0.5945	**0.6115**
yeast	0.7016	0.7089	0.7099	0.6584	**0.7522**
Average value	0.6661	0.6683	0.6698	0.6619	**0.6843**
Average rank	3.80	3.60	3.10	3.30	**1.20**
Adjusted p-value	9.44E-04*	2.10E-03*	0.0072*	0.0060*	–

Table 9: Sensitivity results of decision tree on real datasets.

Dataset	SMOTE3	SMOTE5	SMOTE7	NaNSMOTE	ExNN-SMOTE
satimage	0.6087	0.6058	0.6275	0.6120	**0.6501**
credit-g	0.5087	0.5147	0.5153	0.5220	**0.5267**
ecoli	0.6343	0.6286	0.6400	0.5886	**0.6686**
cmc	0.4366	0.4289	0.4247	0.4397	**0.4595**
cleveland	0.2686	0.3314	0.3486	**0.4000**	0.3314
glass	0.4700	0.4400	0.4533	0.2900	**0.4967**
abalone	0.4967	0.4890	0.5045	0.5266	**0.5499**
solar-flare	0.3317	**0.3711**	0.3394	0.3444	0.3550
transfusion	0.4182	0.4294	0.4249	0.4122	**0.4311**
yeast	0.4600	0.4884	0.4935	0.3567	**0.5655**
Average value	0.4633	0.4727	0.4772	0.4492	**0.5034**
Average rank	3.90	3.55	2.90	3.30	**1.35**
Adjusted p-value	0.0012*	0.0056*	0.0284*	0.0116*	–

Table 10 presents the average numbers of NaNs and ExNNs in each real dataset. It can be observed that the average number of neighbors in NaNSMOTE is close to five, whereas in ExNN-SMOTE, each example has about nine neighbors. We aim to emphasize that ExNN-SMOTE has better results than NaNSMOTE because ExNN-SMOTE uses a more appropriate number of neighbors. This can be partly concluded by comparing with results of SMOTE (k = 3, 5, 7) in Tables 8 and 9.

Table 10: Average numbers of NaNs in NaNSMOTE (M1) and ExNNs in ExNN-SMOTE (M2) on real datasets.

Dataset	M1	M2	Dataset	M1	M2
satimage	5.18	14.40	glass	2.44	5.33
credit-g	5.51	7.20	abalone	6.43	10.67
ecoli	3.34	6.08	solar-flare	3.28	7.45
cmc	5.31	11.51	transfusion	5.35	9.22
cleveland	3.55	6.56	yeast	3.34	7.49
Average	4.37	8.59			

5. Conclusions

In this paper, we propose an oversampling method ExNN-SMOTE to solve the problems of parameter-dependence and overgeneralization in SMOTE. The experiments demonstrate that the proposed ExNN-SMOTE can deal with complicated distribution characteristics (small disjuncts and overlapping between classes) better than the traditional SMOTE and latest NaNSMOTE. This benefits from that ExNN-SMOTE generates more effective borderline examples in the minority class and obtains more explicit decision boundaries.

References

Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Safe-level-SMOTE: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 475–482, Berlin, Heidelberg, 2009. Springer.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

Dongdong Cheng, Qingsheng Zhu, Jinlong Huang, Lijun Yang, and Quanwang Wu. Natural neighbor-based clustering algorithm with local representatives. *Knowledge-Based Systems*, 123:238–253, 2017. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2017.02.027. URL https://www.sciencedirect.com/science/article/pii/S0950705117301119.

Dongdong Cheng, Qingsheng Zhu, Jinlong Huang, Quanwang Wu, and Lijun Yang. A novel cluster validity index based on local cores. *IEEE Transactions on Neural Networks and Learning Systems*, 30(4):985–999, 2019. doi: 10.1109/TNNLS.2018.2853710.

Guodong Du, Jia Zhang, Zhiming Luo, Fenglong Ma, Lei Ma, and Shaozi Li. Joint imbalanced classification and feature selection for hospital readmissions. *Knowledge-Based Systems*, 200:106020, 2020. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2020.106020. URL https://www.sciencedirect.com/science/article/pii/S095070512030318X.

Xin Gao, Bing Ren, Hao Zhang, Bohao Sun, Junliang Li, Jianhang Xu, Yang He, and Kangsheng Li. An ensemble imbalanced classification method based on model dynamic selection driven by data partition hybrid sampling. *Expert Systems with Applications*, 160: 113660, 2020. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2020.113660. URL https://www.sciencedirect.com/science/article/pii/S095741742030484X.

H. Guan, Y. Zhang, M. Xian, · H D Cheng, and · Xianglong Tang. SMOTE-WENN: Solving class imbalance and small sample problems by oversampling and distance scaling. *Applied Intelligence*, 51(4):1394–1409, 2021.

Hui Han, Wenyuan Wang, and Binghuan Mao. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In *International Conference on Intelligent Computing*, pages 878–887, Berlin, Heidelberg, 2005. Springer.

Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge & Data Engineering*, (9):1263–1284, 2008.

Robert C Holte, Liane Acker, Bruce W Porter, et al. Concept learning and the problem of small disjuncts. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, volume 89, pages 813–818, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers.

Jinlong Huang, Qingsheng Zhu, Lijun Yang, DongDong Cheng, and Quanwang Wu. A novel outlier cluster detection algorithm without top-n parameter. *Knowledge-Based Systems*, 121:32–40, 2017. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2017.01.013. URL https://www.sciencedirect.com/science/article/pii/S0950705117300254.

Junnan Li, Qingsheng Zhu, and Quanwang Wu. A self-training method based on density peaks and an extended parameter-free local noise filter for k nearest neighbor. *Knowledge-Based Systems*, 184:104895, 2019. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2019.104895. URL https://www.sciencedirect.com/science/article/pii/S0950705119303582.

Junnan Li, Qingsheng Zhu, Quanwang Wu, and Zhu Fan. A novel oversampling technique for class-imbalanced learning based on SMOTE and natural neighbors. *Information Sciences*, 565:438–455, 2021. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2021.03.041. URL https://www.sciencedirect.com/science/article/pii/S0020025521002863.

Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250:113–141, 2013.

Tomasz Maciejewski and Jerzy Stefanowski. Local neighbourhood extension of SMOTE for mining imbalanced data. In *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 104–111, Washington, DC, USA, 2011. IEEE.

Krystyna Napierala and Jerzy Stefanowski. Types of minority class examples and their influence on learning classifiers from imbalanced data. *Journal of Intelligent Information Systems*, 46(3):563–597, 2016.

Krystyna Napierała, Jerzy Stefanowski, and Szymon Wilk. Learning from imbalanced data in presence of noisy and borderline examples. In *International Conference on Rough Sets and Current Trends in Computing*, pages 158–167, Berlin, Heidelberg, 2010. Springer.

Ronaldo C Prati, Gustavo EAPA Batista, and Maria Carolina Monard. Class imbalances versus class overlapping: an analysis of a learning system behavior. In *Mexican International Conference on Artificial Intelligence*, pages 312–321, Berlin, Heidelberg, 2004. Springer.

José A Sáez, Julián Luengo, Jerzy Stefanowski, and Francisco Herrera. SMOTE-IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. *Information Sciences*, 291:184–203, 2015.

Zian Seng, Sameem Abdul Kareem, and Kasturi Dewi Varathan. A neighborhood undersampling stacked ensemble (NUS-SE) in imbalanced classification. *Expert Systems with Applications*, 168:114246, 2021. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2020.114246. URL https://www.sciencedirect.com/science/article/pii/S0957417420309635.

Jerzy Stefanowski. Overlapping, rare examples and class decomposition in learning classifiers from imbalanced data. In *Emerging Paradigms in Machine Learning*, pages 277–306. Springer, Berlin, Heidelberg, 2013.

Ye Tian, Bo Bian, Xiaofei Tang, and Jing Zhou. A new non-kernel quadratic surface approach for imbalanced data classification in online credit scoring. *Information Sciences*, 563:150–165, 2021. ISSN 0020-0255. doi: https://doi.org/10.1016/j.ins.2021.02.026. URL https://www.sciencedirect.com/science/article/pii/S0020025521001687.

Cunjun Wang, Cun Xin, and Zili Xu. A novel deep metric learning model for imbalanced fault diagnosis and toward open-set classification. *Knowledge-Based Systems*, 220:106925, 2021. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2021.106925. URL https://www.sciencedirect.com/science/article/pii/S095070512100188X.

Lijun Yang, Qingsheng Zhu, Jinlong Huang, Dongdong Cheng, Quanwang Wu, and Xiaolu Hong. Natural neighborhood graph-based instance reduction algorithm without parameters. *Applied Soft Computing*, 70:279–287, 2018. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2018.05.029. URL https://www.sciencedirect.com/science/article/pii/S1568494618303028.

Suwen Zhao and Junnan Li. ELS: A fast parameter-free edition algorithm with natural neighbors-based l sets for k nearest neighbor. *IEEE Access*, 8:123773–123782, 2020. doi: 10.1109/ACCESS.2020.3005815.

Qingsheng Zhu, Ji Feng, and Jinlong Huang. Natural neighbor: A self-adaptive neighborhood method without parameter k. *Pattern Recognition Letters*, 80:30–36, 2016. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2016.05.007. URL https://www.sciencedirect.com/science/article/pii/S016786551630085X.