World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:15, No:3, 2021

Machine Learning Facing Behavioral Noise Problem in an Imbalanced Data Using One Side Behavioral Noise Reduction: Application to a Fraud Detection

Salma El Hajjami, Jamal Malki, Alain Bouju, Mohammed Berrada

Abstract—**With the expansion of machine learning and data mining in the context of Big Data analytics, the common problem that affects data is class imbalance. It refers to an imbalanced distribution of instances belonging to each class. This problem is present in many real world applications such as fraud detection, network intrusion detection, medical diagnostics, etc. In these cases, data instances labeled negatively are significantly more numerous than the instances labeled positively. When this difference is too large, the learning system may face difficulty when tackling this problem, since it is initially designed to work in relatively balanced class distribution scenarios. Another important problem, which usually accompanies these imbalanced data, is the overlapping instances between the two classes. It is commonly referred to as noise or overlapping data. In this article, we propose an approach called: One Side Behavioral Noise Reduction (OSBNR). This approach presents a way to deal with the problem of class imbalance in the presence of a high noise level. OSBNR is based on two steps. Firstly, a cluster analysis is applied to groups similar instances from the minority class into several behavior clusters. Secondly, we select and eliminate the instances of the majority class, considered as behavioral noise, which overlap with behavior clusters of the minority class. The results of experiments carried out on a representative public dataset confirm that the proposed approach is efficient for the treatment of class imbalances in the presence of noise.**

Keywords—**Machine learning, Imbalanced data, Data mining, Big data.**

I. Introduction

THE problems that belong to the class of anomaly detection such as fraud detection, network intrusion detection and medical diagnostics, share a common observation: the captured data is imbalanced. In other words, one of the classes, called minority or postive class, is strongly under-represented compared to the other class, called majority or negative class [1], [2]. The problem with imbalanced data sets is that standard classification learning algorithms assume a relatively uniform distribution of classes. They are often biased towards the majority class and, therefore, there is a higher rate of classification errors for minority class instances [3].

PLAIBDE Project. Programme Opérationnel FEDER-FSE de la région Nouvelle Aquitaine - France, 2017-2020.

Salma El Hajjami, IASSE Laboratory, ENSA-USMBA, Fez, Morocco (corresponding author, e-mail: salma.elhajjami@usmba.ac.ma).
Jamal Malki, L3i Laboratory, La Rochelle University, La Rochelle, France (e-mail: jamal.malki@univ-lr.fr).
Alain Bouju, L3i Laboratory, La Rochelle University, La Rochelle, France (e-mail: alain.bouju@univ-lr.fr).
Mohammed Berrada, IASSE Laboratory, ENSA-USMBA, Fez, Morocco (e-mail: mohammed.berrada@gmail.com).

Unfortunately, the minority class is generally the most important from data analysis point of view. Thus, such a poor classification of minority instances often has serious consequences in real applications. For example, in credit card fraud detection systems, undetected fraudulent transactions are much more serious and costly than detecting normal behavior as fraud. Finding a good solution for imbalanced data with good accuracy has become an important area of research, known as learning from imbalanced data [2], [4], [5].

An important problem, which usually accompanies the imbalanced data classification, is the presence of noise. It is commonly referred to as overlapping data and we named it behavioral noise. The behavioral noise problem occurs when the data instances belonging to a class overlap the data instances of another class. Therefore, the boundaries of the classes may not be clearly defined. This problem plays an important role in the study of imbalanced data. Most classification learning algorithms could lead to classifying minority class instances into majority ones. Thus, classification performance depends on these two main problems: class imbalance and behavioral noise [6].

In this work, we are interested of credit card fraud detection problem. This is a good example of a very imbalanced and overlapping data classification problem [3]. We present a new approach called: One Side Behavioral Noise Reduction (OSBNR) to deal with class imbalance problem, with a particular emphasis on the presence of noise. OSBNR mainly contains two steps. First, a cluster analysis is applied to groups similar instances of the minority class in multiple behavior clusters. Second, we select and eliminate instances of the majority class, considered as behavioral noise, which overlap the behavioral clusters of the minority class.

This article is organized as follows: In Section II, common approaches to address the data imbalance are presented. Section III presents in detail our proposed approach to improve the classification of imbalanced data. While Section IV reports the experiments and shows the results obtained. The final section concludes the paper and provides our insights into future works.

II. Common Approaches for Addressing Data Imbalance

Different approaches have been proposed to deal with imbalanced data problem and improve the performance of prediction [7]-[8]. Often, these approaches are based on either the data level or the algorithm level. Data-level based approaches are independent of the classifier a nd transform the input data distribution to change the balance between classes, for example, using resampling techniques [9], [10]. Algorithmic-level based approaches modify learning or classification strategies in order to pay more attention to the minority class [11], [12].

A. Data-Level Based Approaches

These approaches use sampling methods to produce a well balanced dataset from the imbalanced training dataset. With regard to sampling, we can distinguish two methods: undersampling and oversampling.

1) Undersampling Methods for Data-Level Based Approaches: Eliminate certain majority class instances to achieve a balanced distribution of all classes. However, this can lead to the suppression of informative instances of the majority class, especially if the number of instances of the minority class is very small. Thus, there will be a huge loss of information from majority class instances, leading to a non-optimal classification.

A popular algorithm for undersampling, Condensed Nearest Neighbour Rule (CNN) was proposed in the paper [13]. CNN works by eliminating the majority class samples that are distant from the decision border since these samples can be considered as less relevant for learning. Another popular algorithm for undersampling, Tomek's Link removal (TL) was introduced in [14], This algorithm works by detecting pair of data points, called Tomek's Link, that are each other's nearest neighbor but have different class labels. Undersampling can be done by either removing all Tomek links or by removing the majority class data belonging to the Tomek link. Edited Nearest Neighbor Rule (ENN) was presented in [15]. It removes any instance whose class label is different from the class of, at least, two of its three nearest neighbors. The idea behind this technique is to remove the instances from the majority class near or around the borderline of different classes, in order to increase classification accuracy of minority instances rather than majority instances. Another undersampling technique called Neighbourhood Cleaning Rule (NCR) was proposed by [16]. It uses Wilson's Edited Nearest Neighbour Rule (ENN) [15] to remove instances from the majority class when two out of three of the nearest neighbors of an instance contradict the class. Two improvements to ENN are proposed in [17]: Repeated Edited Nearest Neighnor (RENN) and All-KNN (AKNN). Both methods make multiple passes over the training set repeating ENN. RENN just repeats the ENN algorithm until no further eliminations can be made from the edited set. AKNN repeats ENN for each sample using incrementation values of k each time and removing the sample if its label is not the predominant one at least for one value of k. An undersampling approach called One-Side Selection (OSS) is proposed in [18]. It is the use of Tomek's Link [14] followed by the application of CNN. Tomek's Link is used to remove noisy and borderline majority class examples. Afterwards, the CNN removes examples from the majority class that are distant from decision border. Then a consistent subset of the majority class is formed.

The simplest approach for undersampling is the Random Undersampling (RUS) [19]. RUS selects a subset of the majority class randomly while rejecting the other instances. So, the distribution of classes can be balanced. Several techniques have been proposed to guide the undersampling. In [20], authors present a clustering technique based on undersampling considering the problem of class imbalance for cardiovascular data. The objective was to narrow the gap between the samples of majority class and the samples of minority class. They grouped the samples of the majority class into k clusters. After that, each cluster is combined separately with the samples of the minority class to make k different training datasets. Finally, all the combined datasets are classified with a learning algorithm and the dataset with the greatest precision is used to build the learning model. Two undersampling strategies are introduced in this work [9], they are based on a clustering technique to undersample the majority class in order to produce the same number of data samples as the minority class. In the first strategy, the cluster centers are used to replace the entire majority class dataset. As for the second strategy, the nearest neighbors to each cluster center is used. The work presented in [21] provides an undersampling framework to manage class imbalance in binary datasets by removing potentially overlapped data points. Four methods, based on neighbourhood searching with different criteria, are designed to identify and eliminate majority class instances from the overlapping region.

2) Oversampling Methods for Data-Level Based Approaches: Unlike undersampling, oversampling consists in increasing the number of minority class instances. This can be done in different ways. The most common of these methods is Random Oversampling (ROS), where minority class instances are reproduced to ensure that the two class instances are balanced. The problem is that there is a high probability of overlearning, or overfitting, due to the same instances occurring several times.

To avoid this overlearning problem, a new oversampling method [22] is proposed called Synthetic Minority Oversampling Technique (SMOTE). In SMOTE, the minority class is oversampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. Depending upon the amount of over-sampling required, neighbors from the k-nearest neighbors are randomly chosen. Cluster-based oversampling (CBOS) proposed by [23], first uses the k-means algorithm to group the minority and majority classes separately. All the clusters of the majority class, with the exception of the largest, are randomly oversampled as the same number of training examples as the largest cluster. Then, the total number of majority clusters is equal to each cluster of minority clusters. Authors of [24] proposed a novel ADAptive SYNthetic (ADASYN) sampling approach for learning from imbalanced datasets. ADASYN consists in using a density distribution as a criterion of automatic decision of the number of synthetic samples which must be generated for each minority example, and this by adaptively changing the weights of the different minority examples to compensate for the asymmetric distributions. The modified synthetic minority oversampling technique (MSMOTE) proposed by [25] is the variant of the synthetic minority oversampling technique (SMOTE) which also takes into account the distribution of minority class instances. It divides the data belonging to the minority class into three categories border, noise and latent according to their distance from all the training instances. Another method called Safe-level SMOTE (SSMOTE) have been developed in [26], based on SMOTE. the main objectif of SSMOTE is to avoid the generation of noisy samples. The main objective of SSMOTE is to avoid the generation of noisy samples. Each minority sample is first assigned with a security level value. Then only the samples with higher security level values are selected to generate new minority synthetic samples.The security level value of each minority sample is defined as the number of other minority samples among its k closest neighbors. The study in [10] presents an oversampling method based on k-means clustering and SMOTE (Synthetic Minority Over-sampling Technique), which avoids the generation of noise and overcomes imbalances between and within classes. The results of experiments with 90 datasets show that the training data oversampled with the proposed method improves the results of the classification. In [27], the authors proposed a method of Radial-Based Over-sampling (RBO), which consists in finding regions in which synthetic minority-class objects should be generated on the basis of the imbalance distribution estimation with radial basic functions.

B. Algorithm-Level Based Approaches

Algorithm-level approach adapts existing classification learning algorithms to guide learning towards minority class. It requires a given specific knowledge of the corresponding classifier and of the application field, knowing that in general the classifier fails when the distribution of the classes is uneven.

1) Ensemble Learning for algorithm-Level Based Approaches: Involves improving the performance of single classifiers by inducing several classifiers that may be the same or different and combining them to make a new classifier. Therefore, the basic idea is to build several classifiers from the original data, and then aggregate their predictions when unknown instances are presented. The well-known ensemble methods are bagging and boosting.

Boosting is an iterative algorithm that involves assigning different weights to individuals in the learning dataset. After each iteration, the weight on uncorrectly classified individuals increases and on correctly classified individuals decreases. Since errors are often concentrated on the rare classes, one might think that boosting improves learning on imbalanced datasets by increasing the weights of individuals belonging to the minority class. Different variants of boosting have been proposed to solve data imbalance problems. The AdaBoost algorithm of [28] is the most representative algorithm. It was the first practical boosting algorithm, and remains one of the most widely used and studied, with applications in numerous fields.

In [29], authors proposed a boosting algorithm called: RareBoost. This algorithm applies the following rule: if the number of true positives is greater than the number of false positives, the weight of well-classified individuals decreases. The weight of misclassified individuals increases if the number of true negatives is greater than that false negatives. SMOTEBoost, proposed in [30], combines the SMOTE algorithm with AdaBoost, resulting in a hybrid sampling/boosting algorithm that outperforms both SMOTE and AdaBoost. In [31], a simpler and faster alternative to SMOTEBoost is proposed, which is another algorithm that combines boosting and data sampling called RUSBoost. It is designed to improve the performance of models trained on skewed data.

In [12], the authors present an ensemble algorithm called: EUSBoost based on RUSBoost, which combines random undersampling with boosting algorithm. EUSBoost aims to improve the existing proposals enhancing the performance of the base classifiers by the usage of the evolutionary undersampling approach. This approach promotes diversity favoring the usage of different subsets of majority class instances to train each base classifier.

Bagging also called bootstrap aggregating, is a machine learning ensemble meta-algorithm based on a stochastic process of updating the training dataset to create a diverse set of classifiers. This method consists in constructing each classifier using a sample of instances taken from the original dataset with a replacement. In order to guarantee a sufficient number of instances per classifier, while considering each instance with equal weight, each sample generally contains the same number of instances as in the original dataset. Finally, when an unknown instance is presented to each individual classifier, a majority or weighted vote is used to infer the class.

An overall machine learning method known as Random Forest (RF) was proposed by [32]. It uses several trees as classifiers using bagging. After obtaining the majority vote on all classifiers, the RF method combines information on all trees to reveal varying importance. In [33], authors introduce an ensemble algorithm called: Roughly Balanced (RB) bagging. It uses a novel sampling technique to improve the original bagging algorithm for datasets with imbalanced class distributions. For this sampling method, the number of samples in the largest and smallest classes are different, but they are effectively balanced when averaged over all of the subsets.

In [34], a new type of bagging called Neighbourhood Balanced Bagging (NBBag) is proposed. It is based on a different steps. First, instead of integrating bagging with pre-processing, the standard bagging idea is kept, but sometimes radically. Probabilities of sampling examples to bootstraps are changed by increasing the chance of drawing minority examples. Furthermore, the role of difficult minority examples with respect to the type of their neighbourhood is amplified. The strength of amplification can be parameterized in their setting.

2) Cost-sensitive Learning for lgorithm-Level Based Approaches: These algorithms are sensitive to different costs associated with certain characteristics of the considered problems. Costs can come from various aspects related to a given real problem and can be provided by the field's expert or learned during the training phase of the classifier. Many cost-sensitive methods have been proposed. A general cost-sensitive learning framework, called "MetaCost" is presented in [35]. It begins to learn an internal cost-sensitive model by applying a cost-minimizing procedure, which employs a base learning algorithm. Then, MetaCost estimates class probabilities using bagging and then re-labels the training instances with their minimum expected cost classes. Finally, it relearns a model using the modified training set.

Another cost-sensitive approach called "Adacost" was proposed in [36]. In this approach, examples belonging to the rare class which are misclassified are assigned higher weights than those belonging to the common class. It is empirically demonstrated that the proposed system produces lower cumulative misclassification costs.

In [37], authors study the effect of sampling and threshold-moving in training cost-sensitive neural networks. A threshold-moving technique was used to move the output threshold toward inexpensive classes such that examples with higher costs become harder to be misclassified. In [11], cost-sensitive boosting algorithms to improve the classification of imbalanced data were examined. Authors proposed three cost-sensitive boosting algorithms called AdaC1, AdaC2 and AdaC3 by introducing cost elements as part of learning in AdaBoost's weight update formula.

A wrapper framework incorporating the measurement of evaluation (area under the curve (AUC) and G-mean) in the objective function of cost sensitive Support Vector Machine (SVM) directly is proposed in [38]. The main goal is to improve the performance of the classification by simultaneously optimizing the best pair of subset of functionalities, and misclassification cost parameters.

The authors of [39] studied the class imbalance problem in the context of neural networks using MultiLayer Perceptron (MLP). A cost-sensitive algorithm (CSMLP) is proposed to improve the discriminatory capacity of MLPs (two-class). The CSMLP formulation is based on a common objective function which uses a single cost parameter to distinguish the importance of class errors. Authors of [40] introduced an ensemble of cost-sensitive decision trees for imbalanced classification. It is based on combining cost-sensitive decision trees with random subspace based feature space partitioning.

III. Proposed Approach

Most classification learning algorithms are often biased toward the majority class due to the data imbalance distribution, which leads to a higher misclassification rate for the minority class instances [3]. Unfortunately, the minority class is usually the most important from a data analysis perspective. So, such misclassification of minority instances often has serious consequences in real applications. Another critical factor in real world imbalanced data concerns presence of a relatively large number of noise instances from the majority class located inside the minority class. This problem is commonly referred to as overlapping data and we named it "behavioral noise". The behavioral noise implies that some instances of a class have similar characteristics to those of a different class. The presence of noise has a severe impact in learning problems. The generated models can become more complex, showing less generalization abilities, lower precision, and higher computational cost [41]. So, the classification performance depends on these two main problems: class imbalance and behavioral noise.

In this work, we focus on credit card fraud detection as a very imbalanced and overlapped data classification problem, where non-fraudulent samples are much more numerous than fraud samples [3]. Also, it is known that some users share a behavior where the transactions are similar. While the transaction behavior of some users resembles no behavior and may even behave like transactions unlike their labels. For example, if a user's credit card information is stolen, fraudsters will make several large transactions in a short time to maximize the benefits, while some normal users may also make large transactions in a short time for certain reasons. The normal behavior of an individual user is therefore close to fraud. We define this type of transaction as behavioral noise. The existence of behavioral noise pushes the system to judge certain fraudulent transactions as authentic transactions. This leads to an erroneous classification which can be costly. Undetected fraudulent transactions (false positive) are much more serious and costly than detecting normal behavior as fraud (false negative). The cost of false positives is financial in nature, it varies according to the amount of the transaction. On the other hand, the cost of false negatives is measured in terms of customer dissatisfaction, and this latter can be resolved by strategies to compensate and retain customers.

The main objective of our approach, called "One Side Behavioral Noise Reduction" (OSBNR) is to handle behavioral noise to improve the classification of the minority class instances. OSBNR consists of separating normal transactions (majority class instances) from fraudulent ones (minority class instances). Then, a cluster analysis is applied to group similar instances of the minority class containing the fraudulent transactions in several subsets, that form several behavior groups. The second step eliminates normal transactions behaviors, considered as behavioral noise, which overlap with the fraudulent transactions behaviors.

Fig 1 shows the main steps of OSBNR approach:

1) Separate: separates the majority Dmj and minority Dmn from the original training dataset.
2) Clustering: using k-means clustering algorithm [42] to form samples of similar Dmn instances into a number

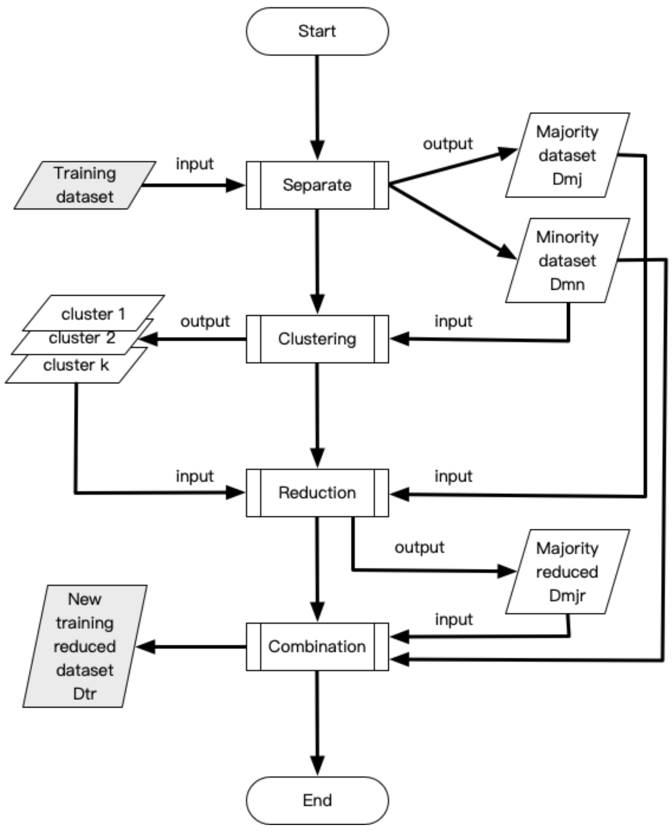

Fig. 1 Flowchart of the OSBNR approach

of behavior clusters. Each cluster seems to have distinct characteristics in the high-dimensional features space.

3) Reduction: carries out the behavioral noise reduction of the majority instances with those of the minority class. The Euclidean distance is used to measure the level of similarity between the different clusters centers of minority class and the majority class instances. So first, we calculate the furthest distance $dmax_i, 1 < i < k$, from minority instances to the cluster center $Cmin_i$ for each minority cluster according. Second, the distances $dmaj_{ij}$ between majority instances and different clusters centers of minority class C_i are obtained. As a result, all majority instances in the minority clusters area are identified if $dmax_i \geq dmaj_{ij}$. So, they are considered as noisy instances and then eliminated.
4) Combination: we combine the reduced majority instances set $Dmjr$ with the minority instances set Dmn to have a new training dataset Dtr.

Accurate identification and elimination of these instances maximize the visibility of the minority class instances and at the same time minimize excessive elimination of data.

IV. Experiments and Results

A. Dataset Description

For this work, we use the Kaggle credit card fraud detection dataset [43]. It contains transactions made by credit card during two days of September 2013 by European card holders. Table I provides statistics for the dataset and shows that the minority class (fraud) accounts for 0.172% of all transactions. Therefore, this dataset is highly imbalanced [3]. It contains 31 numerical features. Since some of the input features contains financial information, the PCA transformation of 28 digital input features (named $V_1, \ldots, V_{28}$) were performed due to confidentiality issues. Three of the given features weren't transformed. Time feature shows the time between first transaction and every other transaction in the dataset. Amount feature is the amount's value spent in a single transaction made by credit card. Class feature represents the label, and takes only 2 values: value 1 in case of fraud transaction and 0 otherwise.

TABLE I
Kaggle Credit Card Fraud Dataset Details

Transactions	Majority class	Minority class	Columns
284 807	284 315	492	31

B. Feature Selection

Feature selection is a fundamental technique that selects the most relevant features from the given dataset. Choosing the right features wisely and removing the less important ones can reduce over-learning, improve accuracy, and reduce training time. Visualization techniques can be helpful in this process. Formally, we select a subset of features or attributes from the set of features and eliminate redundant features that do not contribute to performance. Thus, a feature is important when its data distributions of the two classes are divergent. Therefore, this feature can potentially separate the two classes and improve prediction performance.

Fig. 2 shows the class distribution for some features of our dataset. We can see for V_9, V_{10}, V_{11}, V_{12} and V_{14} a significant divergence of class distribution. They are therefore features with a strong predictive power. So, we can keep them during the models construction. Similarly, we can see for feature V_{13} that the distribution of normal transactions (majority class) corresponds to the distribution of fraudulent transactions (minority class). This feature cannot effectively contribute to the separation between the two classes. We carried out this process for all 28 features. As a result, 11 relevant features were selected for our experiments: V_3, V_4, V_9, V_{10}, V_{11}, V_{12}, V_{14}, V_{16}, V_{17}, V_{18} and V_{19}.

C. Classifiers and Resampling Techniques

In this work, we applied various resampling techniques such as the Condensed Nearest Neighbour Rule (CNN) [13], Tomek's links (TL) [14], One-Side Selection (OSS) [18], Edited Nearest Neighbour Rule (ENN) [15], Repeated Edited Nearest Neighbor (RENN) [17], All-KNN (AKNN) [17] and Neighbor Cleaning Rule (NCR) [16]. We evaluated their performance with the proposed approach OSBNR using the best and widely used classifiers: Random Forest (RF) and Multilayer Perceptron (MLP) [44]:

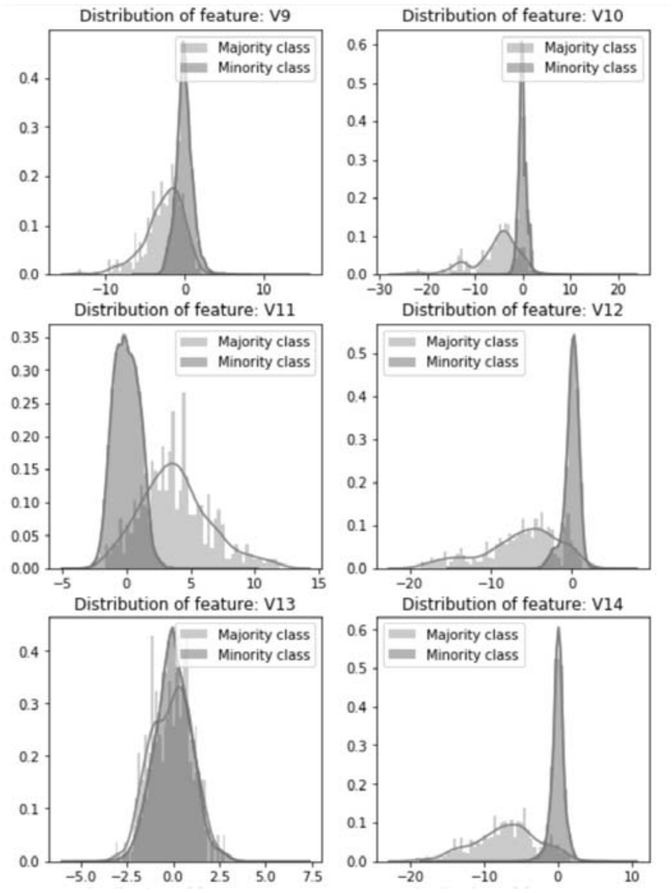

Fig. 2 Class distribution histogram on some features

- Random forest (RF) is an algorithm that can be used in both classification and regression problems. It consists of many decision trees. This algorithm works best when there are more trees in the forest and prevents the model from over-adapting. Each decision tree in the forest gives results. These results are merged in order to obtain a more precise and stable prediction [32].
- Multilayer perceptron (MLP) is an artificial neural network with direct action which is made up of at least 3 layers of nodes: entry layer, hidden layer and exit layer. Each node uses an activation function. The activation function calculates the weighted sum of its inputs and adds a bias. This allows us to decide which neuron should be removed and not taken into account in the external connections.

RF and MLP models parameters were determined from various preliminary tests carried out on the training data, as shown in Table II.

D. Evaluation Metrics

Evaluation metrics play an important role to assess and guide learning algorithms [7]. The common metric used is accuracy. However, accuracy is not a good indicator of the actual classification performance when the class distribution is not uniform, especially for the positive (minority) class. Indeed, because it has less effect on accuracy compared to the negative (majority) class. As in [45], we consider other metrics summarized as follows, where:

TABLE II
RF AND MLP PARAMETERS USED

Classifiers	Parameter
Random forest (RF)	Number of trees = 20 Depth of each tree = 8 Impurity = Gini
Multilayer perceptron (MLP)	Number of iterations = 100 Tolerance parameter = 1e-6

$$\begin{cases} FP & \text{false positive} \\ FN & \text{false negative} \\ TP & \text{true positive} \\ TN & \text{true negative} \end{cases}$$

- Precision or Positive Predictive Value (1): represents the proportion of positive samples that were correctly classified to the total number of positive predicted samples.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

- True Positive Rate (2): called Sensitivity or Recall, is the number of actual positives which are predicted positives.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

- F1-score or F-measure (3): represents the harmonic mean of precision and recall. The value ranges from 0 to 1, if the value is high then F1-score indicates high classification performance.

$$F1\text{-score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3}$$

- AUC (4): represents the ability to distinguish classes, which considers both the true positive rate TPR (2) and the false positive rate FPR (5). AUC is based on the consideration that the higher the true positive rate TPR is, and the lower false positive rate FPR is, classification performance is better.

$$AUC = \frac{1 + TPR - FPR}{2} \tag{4}$$

Where False Positive Rate (FPR (5)) represents the proportion of legitimate samples that were wrongly predicted as fraud.

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

V. Results Analysis

A. Training and Test Datasets Used for the Experiments

We present different experiments to compare the performance of our proposed OSBNR approach and the state-of-art resampling methods (CNN, ENN, AKNN, RENN, TL, OSS, and NCR). As there is no rule-of-thumb for how to divide a dataset into training and test sets, we have noticed that in the case of the $70/30$ rule, the percentages of the minority class of the training and test sets are: 80%, 20% respectively of the total fraud. In order to demonstrate the effectiveness of the proposed OSBNR method to deal with the imbalance and overlapping between classes problem, we studied 3 different divisions of the dataset by resampling the minority class based on the ratios: $80/20, 70/30, 60/40$ and the majority class based on the $70/30$ ratio. Table III presents training and test datasets used as input of our OSBNR approach (Fig. 1).

TABLE III
TRAINING AND TEST DATASETS USED FOR THE EXPERIMENTS

		Total	**Majority class**	**Minority class**
100%	**Dataset**	284 807	284315 - 100%	492 - 100%
Rule 80/20	**Training**	199 413	199020 - 70%	393 - **80%**
	Test	85 394	85295 - 30%	99 - **20%**
Rule 70/30	**Training**	199 364	199020 - 70%	344 - **70%**
	Test	85 443	85295 - 30%	148 - **30%**
Rule 60/40	**Training**	199 315	199020 - 70%	295 - **60%**
	Test	85 492	85295 - 30%	197 - **40%**

B. *Performance Study of the OSBNR: Case of All Features*

In this section, we analyze the impact of the proposed OSBNR approach on the performance of each classifier by comparing it with existing resampling methods taking into account all the features and according to 3 different divisions of the dataset. The results are calculated for four metrics: AUC, Precision, Recall and F1-score.

Figs. 3 and 4 show the results using the AUC metric for the RF and MLP classifiers respectively. The most interesting observation is that the proposed OSBNR offers significantly better performance compared to the other methods for the two classifiers from the AUC point of view for all the distributions of training and test sets.

For the RF classifier, the best score is obtained by RF_OSBNR with an AUC value of ($AUC = 0.9341$), RF_CNN takes second place with a score of ($AUC = 0.9079$), when the training and test sets are set to $80/20$ rule. For the MLP classifier, the best AUC score rule is obtained by MLP_OSBNR with a value of ($AUC = 0.9487$), followed by MLP_OSS with a score of ($AUC = 0.9079$) when the training and test sets are set to $80/20$ rule.

Similarly, Figs. 5 and 6 present the results in terms of precision. The results illustrated in Fig. 5 show that OSBNR outperforms the other resampling methods for all the distributions of the training and test sets. The best precision score for the RF classifier is obtained by the OSBNR approach when the training and test sets are set at 80% and 20% fraud with a score of ($Precision = 0.8686$), while RF_CNN takes second place with a score of ($Precision = 0.8163$).

Similar in MLP, as illustrated in Fig. 6, it is clear that the best precision score is obtained by MLP_OSBNR with a score of ($Precision = 0.8979$), followed by MLP_OSS with a score of ($Precision = 0.8511$). Based on these results, we

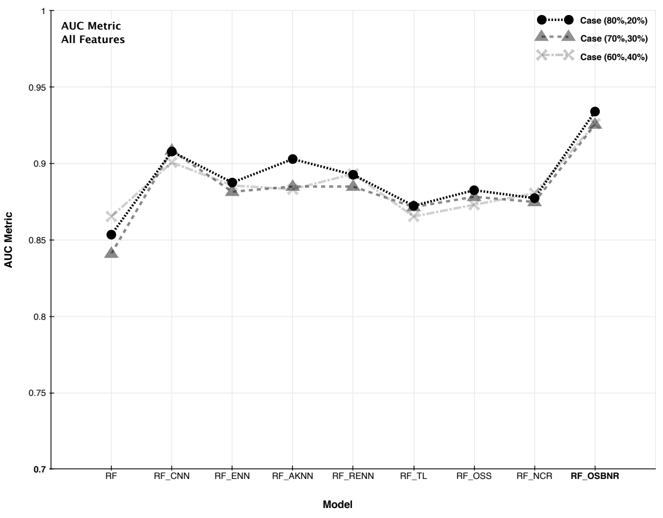

Fig. 3 AUC metric for OSBNR and the reference resampling approaches: RF as base classifier case of all features

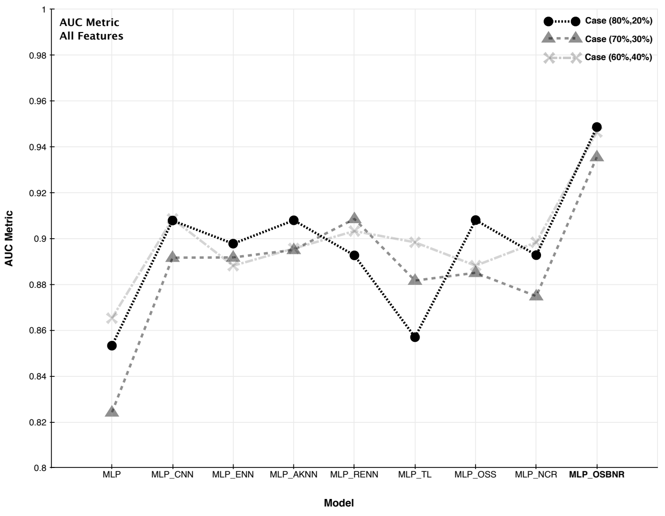

Fig. 4 AUC metric for OSBNR and the reference resampling approaches: MLP as base classifier case of all features

conclude that the proposed OSBNR can significantly improve the recognition rate of minority samples.

Figs. 7 and 8 show the results obtained using Recall measure. The interesting observation is that the RF and MLP classifiers without preprocessing clearly offer the best performance in terms of recall metric. Such a result was expected in a way, because the resampling methods were introduced to manage class imbalance and class overlap problems.They eliminate the instances of the majority class considered as noise to improve the prediction of instances of the minority class, and this can slightly increase the rate of false negatives. For the RF classifier, the second best recall score is obtained by RF_OSS with a value of ($Recall = 0.96$) when the training and test sets are set at 60% and 40% fraud. Similarly, for MLP the second place is occupied by MLP_OSS

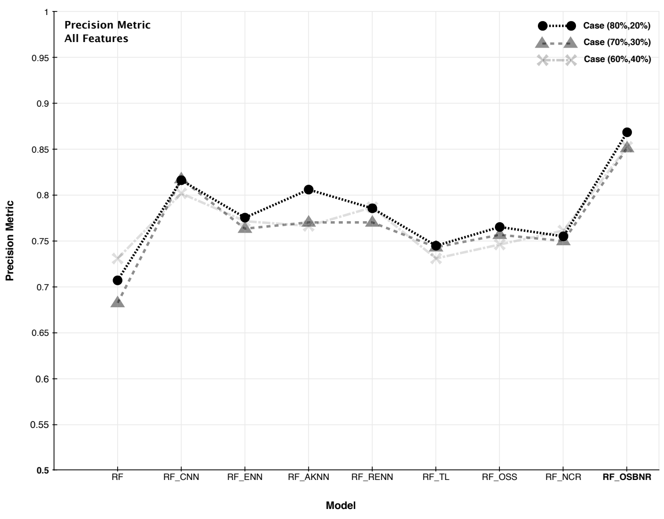

Fig. 5 Precision metric for OSBNR and the reference resampling approaches: RF as base classifier case of all features

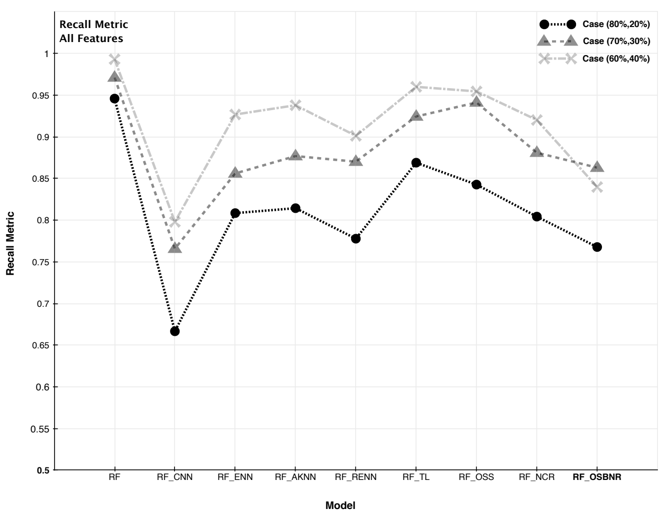

Fig. 7 Recall metric for OSBNR and the reference resampling approaches: RF as base classifier case of all features

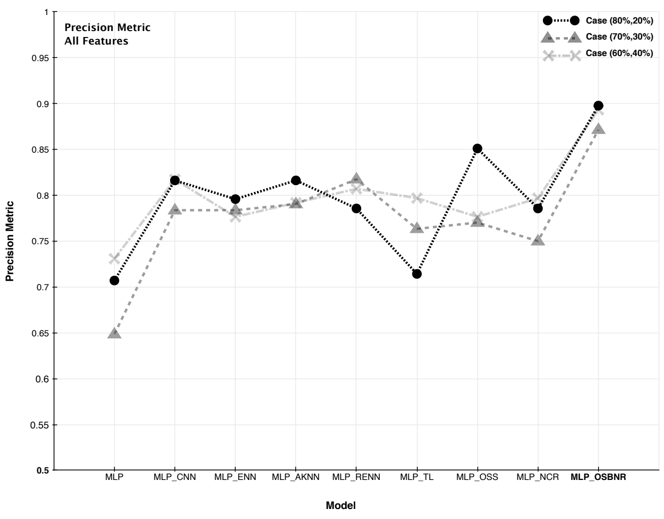

Fig. 6 Precision metric for OSBNR and the reference resampling approaches: MLP as base classifier case of features

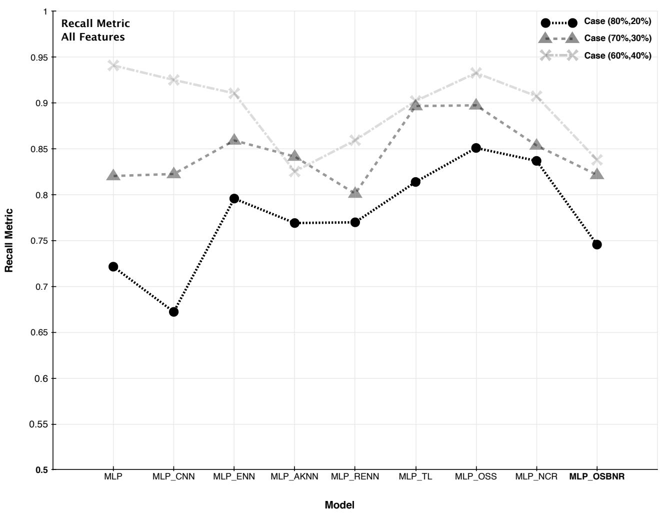

Fig. 8 Recall metric for OSBNR and the reference resampling approaches: MLP as base classifier case of all features

with a score of $(Recall = 0.9329)$ with the $60/40$ rule.

As for measure F1-score, Figs. 9 and 10 present the results of all the resampling methods by applying the two learning classifiers. We see that the OSBNR approach outperforms the other resampling methods for all the distributions of the training and test sets for two classifiers. For RF, the best F1-score is obtained by RF_OSBNR with a score of $(F1 - score = 0.8572)$ according to the $70/30$ rule, followed by RF_AKNN with a score of $(F1 - score = 0.8436)$ when applying rule $60/40$ to divide the training and test sets. Regarding MLP, the best score is obtained by MLP_CNN with a F1-score of 0.8679 when the training and test sets are set at 60% and 40% fraud, MLP_OSBNR takes second place with a value of 0.8648.

C. Performance Study of the OSBNR: Case of Relevant Features

In this section, we analyze the impact of the proposed OSBNR on the performance of each classifier by comparing it with existing resampling methods taking into account relevant features and also according to the 3 different divisions of the dataset. The results are calculated for four metrics: AUC, Precision, Recall and F1-score.

In order to better situate the results, we start by reporting on AUC metric. From Figs. 11 and 12, we can see that the best AUC scores are obtained by RF and MLP combined with OSBNR for all the distributions of the sets of training and testing. For the RF classifier, the best score is obtained by RF_OSBNR with an AUC value of $(AUC = 0.9442)$, RF_CNN takes second place with a score of $(AUC = 0.9129)$

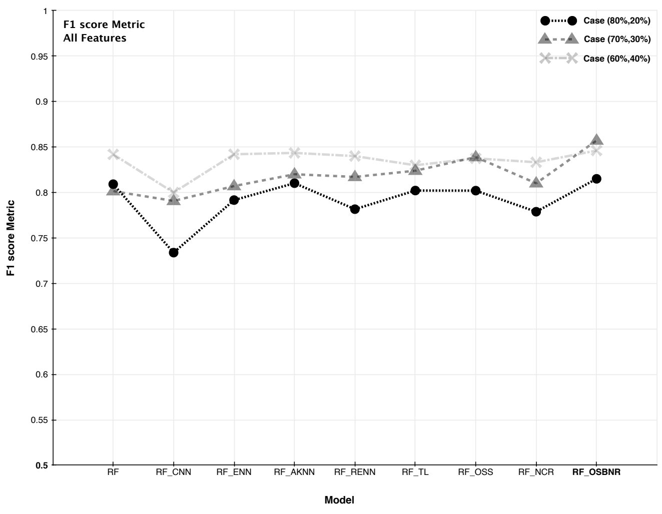

Fig. 9 F1-score metric for OSBNR and the reference resampling approaches: RF as base classifier case of all features

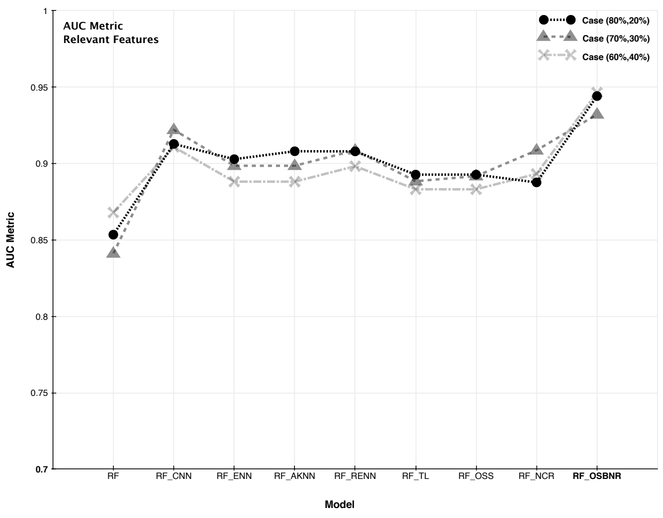

Fig. 11 AUC metric for OSBNR and the reference resampling approaches: RF as base classifier case of relevant features

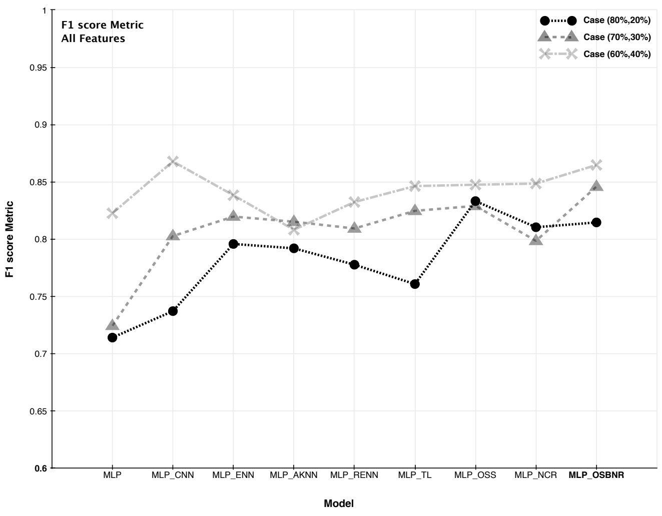

Fig. 10 F1-score metric for OSBNR and the reference resampling approaches: MLP as base classifier case of all features

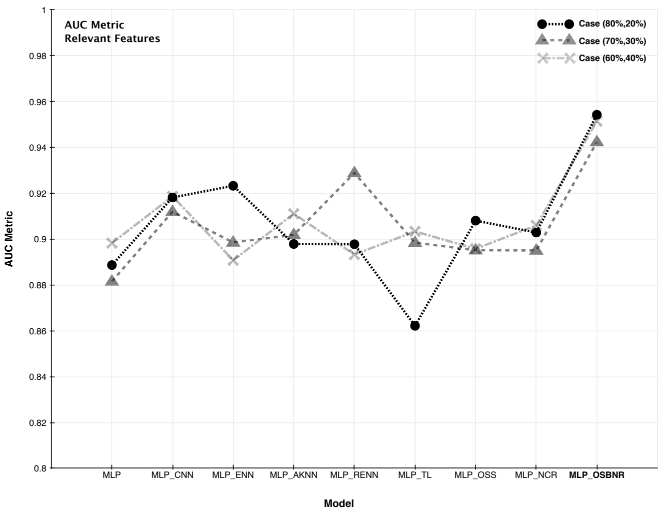

Fig. 12 AUC metric for OSBNR and the reference resampling approaches: MLP as base classifier case of relevant features

according to the $80/20$ rule. Regarding the MLP classifier, the best AUC score is obtained by MLP combined with OSBNR when applying the $80/20$ rule with a value of ($AUC = 0.9543$), followed by MLP_RENN with a score of ($AUC = 0.9289$), while MLP_OSS maintained its score with the same value of ($AUC = 0.9079$).

From these results, we can conclude that after eliminating redundant features that do not contribute to performance, we get continuity or even improvement in predictive performance.

Similarly, Figs. 13 and 14 present the results in terms of precision. The results illustrated in Fig. 13 show that the best precision score for the RF classifier is obtained by the OSBNR approach for all the distributions of training and test sets with a best value score of ($Precision = 0.8934$) according to $80/20$ rule, which means that this model offers a better prediction of minority instances.

Similar in MLP, As illustrated in Fig. 14, it is clear that the best precision score is obtained by MLP_OSBNR with a best score of ($Precision = 0.909$).

With regard to the Recall measure, Figs. 15 and 16 show the results obtained. we also notice that the RF and MLP classifiers without preprocessing clearly offer the best performance in terms of recall metric. For the RF classifier, the second best recall score is obtained by RF_OSS with a value of ($Recall = 0.928$) when the training and test sets are set at 70% and 30% fraud. Similarly, for MLP the second place is occupied by MLP_AKNN with a score of ($Recall = 0.9474$) according to the $70/30$ rule.

As for measure F1-score, Figs. 15 and 16 show the results of all the resampling methods by applying the two learning classifiers. For RF, we see that the OSBNR outperforms

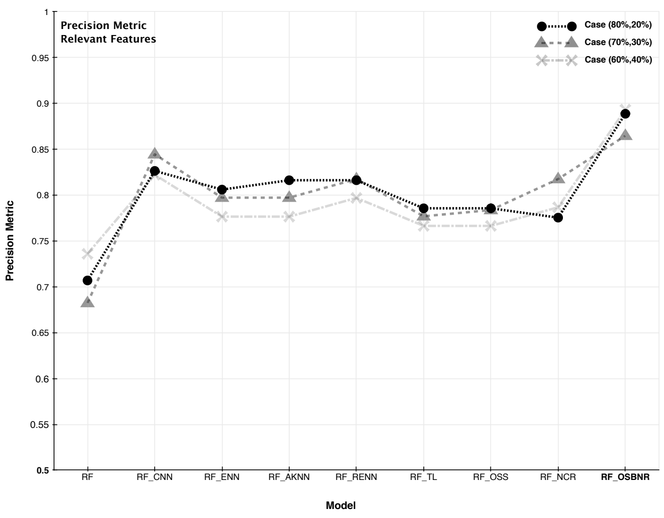

Fig. 13 Precision metric for OSBNR and the reference resampling approaches: RF as base classifier case of relevant features

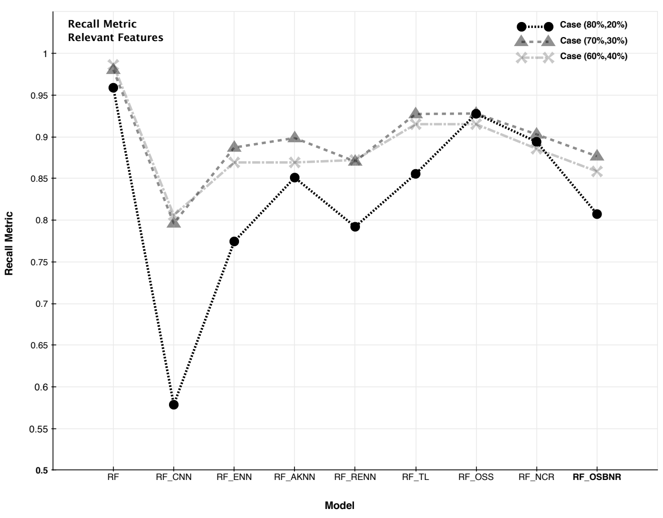

Fig. 15 Recall metric for OSBNR and the reference resampling approaches: RF as base classifier case of relevant features

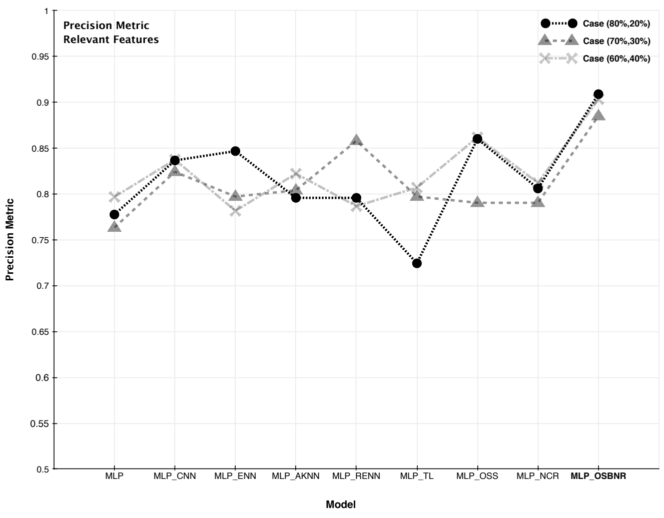

Fig. 14 Precision metric for OSBNR and the reference resampling approaches: MLP as base classifier case of relevant features

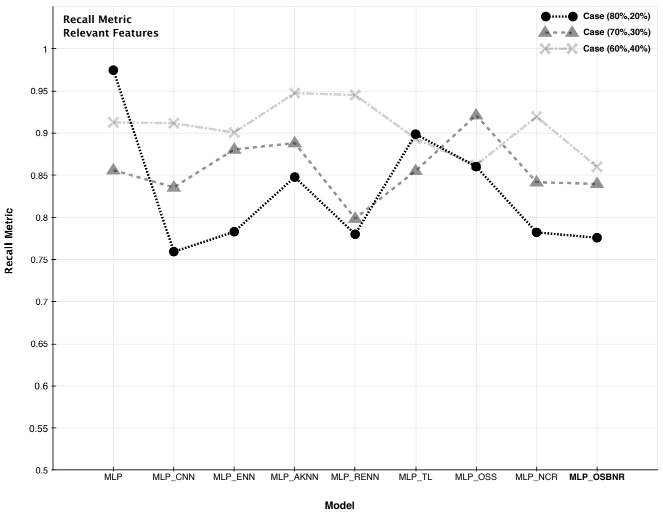

Fig. 16 Recall metric for OSBNR and the reference resampling approaches: MLP as base classifier case of relevant features

the other resampling methods for all the distributions of the training and test sets. Likewise, for MLP, the best score is obtained by MLP combined with OSBNR for all the distributions of training and test sets.

D. Performance of the Re-sampling Methods in Terms of Processing Time

The time processing for all the compared methods is presented in Fig. 19. It shows the average execution time for the different resampling methods on the used dataset considering all the features. From the results, we can see that OSBNR is the fastest among all the comparative resampling techniques and OSS ranks second. TL, ENN, NCL are moderately fast for processing. We notice that the CNN method is more complex because it takes a certain significant operating time. Therefore, we conclude that a combination approach of OSBNR would be the optimal choice, offering the best classification performance and requiring the least processing time.

VI. Conclusion and Future Work

In this paper, we provide a survey of the existing methods for solving the two-class imbalanced classification problem. Then we present a new approach called: One Side Behavioral Noise Reduction (OSBNR). Our approach combines clustering analysis and a behavioral noisy data reduction process.

To study the effectiveness of the proposed method, we compare it to several state-of-the-art resampling methods. Two learning classifier, namely Random Forest and MultiLayer Perceptron, have been tested over these resampling methods. Experimental results measured using four metrics (AUC, Precision, Recall, F1-score) indicate that OSBNR achieves

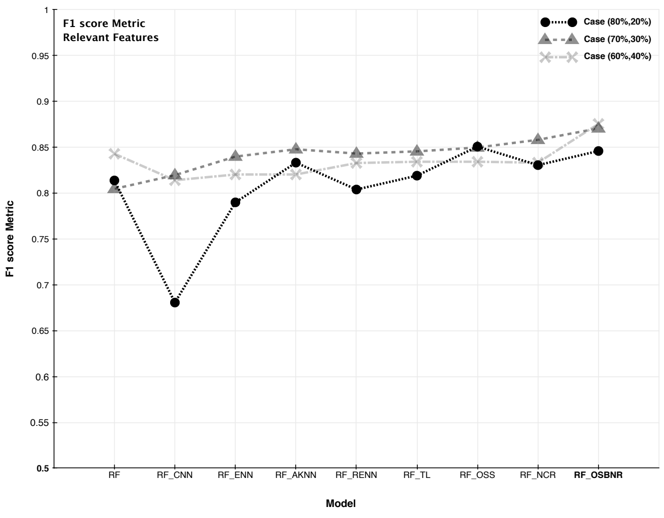

Fig. 17 F1-score metric for OSBNR and the reference resampling approaches: RF as base classifier case of relevant features

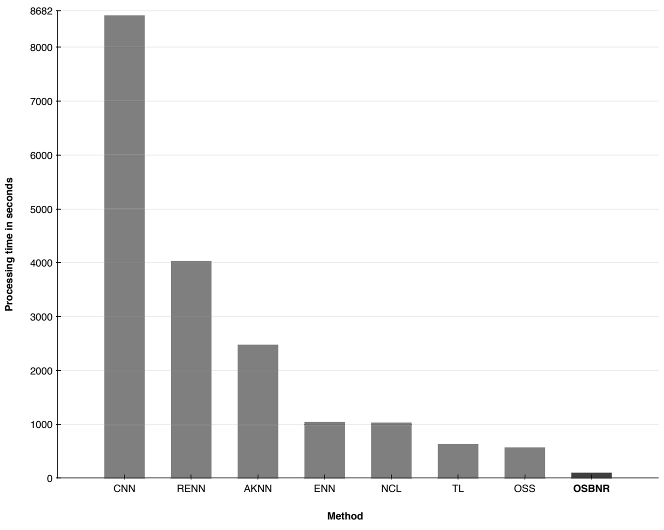

Fig. 19 Performance of the resampling methods: time processing

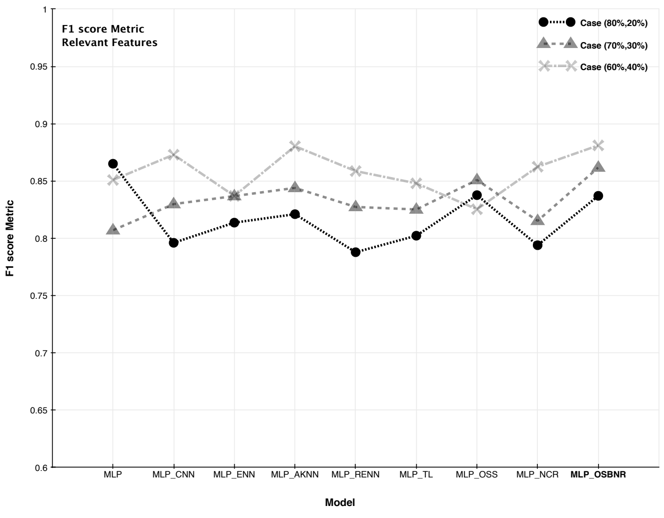

Fig. 18 F1-score metric for OSBNR and the reference resampling approaches: MLP as base classifier case of relevant features

much better classification performance than the other compared methods to deal with noise data problem with a significant difference.

This work constitutes an important part of the framework in development. Thus, we wish to study the behavior of the scaling of our approach in the context of a real application. This will raise two fundamental questions:

- Confidence and predictability of predictions for decision making. The main objective of our explanatory approach to machine learning is to propose methods to understand and explain how the system produces its decisions in case of real domain application.
- Notion of uncertainty in machine learning which is of major importance and constitutes a key element of modern machine learning methodology. It has gained in importance due to the increasing relevance of machine learning in real applications.

Acknowledgment

This work is carried out thanks to the support of the European Union through the PLAIBDE project of the FEDER-FSE operational program for the Nouvelle-Aquitaine region, France. The project is supported by aYaline company, with partners: LIAS-ENSMA laboratory in Poitiers, and the L3i laboratory at La Rochelle University.

References

[1] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002.
[2] A. Ali, S. M. Shamsuddin, and A. Ralescu, "Classification with class imbalance problem: a review," *Int. J. Advance Soft Compu. Appl*, vol. 7, no. 3, pp. 176–204, 2015.
[3] A. Dal Pozzolo, O. Caelen, R. A. Johnson, and G. Bontempi, "Calibrating probability with undersampling for unbalanced classification," in *2015 IEEE Symposium Series on Computational Intelligence*, pp. 159–166, IEEE, 2015.
[4] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 221–232, 2016.
[5] H. Alberto Fernández, L. Salvador García, G. Mikel, C. P. Ronaldo, K. Bartosz, and H. Francisco, *Learning from imbalanced data sets*. Springer International Publishing, 2018.
[6] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Folleco, "An empirical study of the classification performance of learners on imbalanced and noisy software quality data," *Information Sciences*, vol. 259, pp. 571–595, 2014.
[7] Y. Sun, A. K. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *International journal of pattern recognition and artificial intelligence*, vol. 23, no. 04, pp. 687–719, 2009.
[8] Y. Sui, M. Yu, H. Hong, and X. Pan, "Learning from imbalanced data: A comparative study," in *International Symposium on Security and Privacy in Social Networks and Big Data*, pp. 264–274, Springer, 2019.
[9] W.-C. Lin, C.-F. Tsai, Y.-H. Hu, and J.-S. Jhang, "Clustering-based undersampling in class-imbalanced data," *Information Sciences*, vol. 409, pp. 17–26, 2017.
[10] G. Douzas, F. Bacao, and F. Last, "Improving imbalanced learning through a heuristic oversampling method based on k-means and smote," *Information Sciences*, vol. 465, pp. 1–20, 2018.

[11] Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang, “Cost-sensitive boosting for classification of imbalanced data,” *Pattern Recognition*, vol. 40, no. 12, pp. 3358–3378, 2007.

[12] M. Galar, A. Fernández, E. Barrenechea, and F. Herrera, “Eusboost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling,” *Pattern recognition*, vol. 46, no. 12, pp. 3460–3471, 2013.

[13] P. Hart, “The condensed nearest neighbor rule (corresp.),” *IEEE transactions on information theory*, vol. 14, no. 3, pp. 515–516, 1968.

[14] I. Tomek, “Two modifications of cnn,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 7(2), pp. 679–772, 1976.

[15] D. L. Wilson, “Asymptotic properties of nearest neighbor rules using edited data,” *IEEE Transactions on Systems, Man, and Cybernetics*, no. 3, pp. 408–421, 1972.

[16] J. Laurikkala, “Improving identification of difficult small classes by balancing class distribution,” in *Conference on Artificial Intelligence in Medicine in Europe*, pp. 63–66, Springer, 2001.

[17] I. Tomek, “An experiment with the edited nearest-neighbor rule.,” 1976.

[18] M. Kubat and S. Matwin, “Addressing the curse of imbalanced training sets: one-sided selection,” in *Icml*, vol. 97, pp. 179–186, Nashville, USA, 1997.

[19] H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[20] M. Rahman and D. N. Davis, “Cluster based under-sampling for unbalanced cardiovascular data,” in *Proceedings of the World Congress on Engineering*, vol. 3, pp. 3–5, 2013.

[21] P. Vuttipittayamongkol and E. Elyan, “Neighbourhood-based undersampling approach for handling imbalanced and overlapped data,” *Information Sciences*, vol. 509, pp. 47–70, 2020.

[22] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[23] T. Jo and N. Japkowicz, “Class imbalances versus small disjuncts,” *ACM Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 40–49, 2004.

[24] H. He, Y. Bai, E. A. Garcia, and S. 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.

[25] S. Hu, Y. Liang, L. Ma, and Y. He, “Msmote: Improving classification performance when training data is imbalanced,” in *2009 second international workshop on computer science and engineering*, vol. 2, pp. 13–17, IEEE, 2009.

[26] C. Bunkhumpornpat, K. Sinapiromsaran, and C. 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*, pp. 475–482, Springer, 2009.

[27] M. Koziarski, B. Krawczyk, and M. Woźniak, “Radial-based oversampling for noisy imbalanced data classification,” *Neurocomputing*, vol. 343, pp. 19–33, 2019.

[28] Y. Freund and R. E. Schapire, “Schapire R: Experiments with a new boosting algorithm,” in *In: Thirteenth International Conference on ML*, Citeseer, 1996.

[29] M. V. Joshi, V. Kumar, and R. C. Agarwal, “Evaluating boosting algorithms to classify rare classes: Comparison and improvements,” in *Proceedings 2001 IEEE International Conference on Data Mining*, pp. 257–264, IEEE, 2001.

[30] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, “Smoteboost: Improving prediction of the minority class in boosting,” in *European conference on principles of data mining and knowledge discovery*, pp. 107–119, Springer, 2003.

[31] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, “Rusboost: A hybrid approach to alleviating class imbalance,” *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 40, no. 1, pp. 185–197, 2009.

[32] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[33] S. Hido, H. Kashima, and Y. Takahashi, “Roughly balanced bagging for imbalanced data,” *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 2, no. 5-6, pp. 412–426, 2009.

[34] J. Błaszczyński and J. Stefanowski, “Neighbourhood sampling in bagging for imbalanced data,” *Neurocomputing*, vol. 150, pp. 529–542, 2015.

[35] P. Domingos, “Metacost: A general method for making classifiers cost-sensitive,” in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 155–164, 1999.

[36] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan, “Adacost: misclassification cost-sensitive boosting,” in *Icml*, vol. 99, pp. 97–105, 1999.

[37] Z.-H. Zhou and X.-Y. Liu, “Training cost-sensitive neural networks with methods addressing the class imbalance problem,” *IEEE Transactions on knowledge and data engineering*, vol. 18, no. 1, pp. 63–77, 2005.

[38] P. Cao, D. Zhao, and O. Zaiane, “An optimized cost-sensitive svm for imbalanced data learning,” in *Pacific-Asia conference on knowledge discovery and data mining*, pp. 280–292, Springer, 2013.

[39] C. L. Castro and A. P. Braga, “Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data,” *IEEE transactions on neural networks and learning systems*, vol. 24, no. 6, pp. 888–899, 2013.

[40] B. Krawczyk, M. Woźniak, and G. Schaefer, “Cost-sensitive decision tree ensembles for effective imbalanced classification,” *Applied Soft Computing*, vol. 14, pp. 554–562, 2014.

[41] X. Zhu and X. Wu, “Class noise vs. attribute noise: A quantitative study,” *Artificial intelligence review*, vol. 22, no. 3, pp. 177–210, 2004.

[42] K. Alsabti, S. Ranka, and V. Singh, “An efficient k-means clustering algorithm,” *Electrical Engineering and Computer Science. 43. College of Engineering and Computer Science at SURFACE*, 1997.

[43] K. Inc, “Credit card fraud detection: Anonymized credit card transactions labeled as fraudulent or genuine,” 2013.

[44] E. H. Salma, M. Jamal, B. Mohammed, and F. Bouziane, “Machine learning for anomaly detection. performance study considering anomaly distribution in an imbalanced dataset,” in *2020 5th International Conference on Cloud Computing and Artificial Intelligence Technologies and Applications (Cloudtech’20)*, IEEE, 2020.

[45] C. Priscilla and D. Prabha, “Credit card fraud detection: A systematic review,” in *International Conference on Information, Communication and Computing Technology*, pp. 290–303, Springer, 2019.