IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 11, 2019, accepted August 27, 2019, date of publication September 5, 2019,
date of current version September 19, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2939755

An Effective Recursive Technique for Multi-Class Classification and Regression for Imbalanced Data

TAHIRA ALAM, CHOWDHURY FARHAN AHMED, SABIT ANWAR ZAHIN, MUHAMMAD ASIF HOSSAIN KHAN, AND MALIHA TASHFIA ISLAM
Department of Computer Science and Engineering, University of Dhaka, Dhaka 1000, Bangladesh
Corresponding author: Tahira Alam (tahiradu@gmail.com)

ABSTRACT In machine learning, classification and regression are two of the most noteworthy key topics since they occur extensively in numerous real-world applications. However, real life data is hardly ever found balanced, rather skewed data is the common occurrence. This poses some serious challenges to the standard techniques of classification and regression. The performance and effectiveness of these techniques are substantially affected by overfitting, creating a bias towards the majority class. In recent years, quite a few number of methods have been introduced for classification of imbalanced data. But most of them are designed for binary classes and it is difficult or inefficient to extend them for multiple classes. Moreover, data imbalance problem occurs frequently in regression analysis too, with only a handful of algorithms robust enough to tackle to this problem. In this paper, we propose an effective recursive method for multi-class classification with imbalanced data. Our proposed algorithm partitions and balances the data, and is applied recursively coupled with ensemble techniques. Furthermore, we also extend our proposed method to solve the data imbalance problem in regression analysis. Experimental results demonstrate that the proposed recursive technique is effective and improves the performance when compared to existing methods for classification and regression with imbalanced distribution.

INDEX TERMS Classification, imbalance problem, multi-class classification, regression.

I. INTRODUCTION

In machine learning, classification [3], [4] plays a vital role by building models that describe important data classes. Classification usually employs the supervised or unsupervised methods for learning. Supervised classification is a two step process. The first step is a learning phase, and consists of training a classifier based on a predetermined set of input data called the training set. The training dataset is made up of a database of tuples and their associated class labels, where the class labels are discrete and each value denotes a class or category. The second step consists of utilizing the training learnt in the previous step and predict class labels for an unknown test data using the classification model [1].

When the class values are not categorical, but numeric, it is known as regression analysis. Like the classification problem, we can train our model based on a collected training data, and the machine learning algorithm fits a function to this data and predicts numeric values for new test data [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Berdakh Abibullaev.

The imbalance problem is considered as one of the top ten challenging problems in data mining [6]. Most of the traditional classification algorithms assume that the sample distribution among various classes is balanced. But a wide variety of real world data suffer from class imbalance problem. The class imbalance problem occurs when any minority class is outnumbered significantly by majority classes. For the binary class imbalance problem, there are exactly one minority class and one majority class. On the other hand, in the multi-class imbalance problem, there can be several instances of both minority and majority classes. A number of approaches [7], [8] have been proposed for solving the class imbalance problem, but most of them are not adequate enough to be used as state of the art. In addition, data imbalance problem is also present in regression analysis with very few works addressing the issue. Real-life data usually follows normal distribution causing imbalance in the data. For solving the multi-class imbalanced classification problem, we have converted the imbalance problem into a number of balanced problems. We have applied unique recursive based data partitioning techniques and extended our algorithm with an effective recursive approach for solving the data imbalance

regression problem. To the extent of our knowledge no such recursion based method has been proposed for addressing this imbalance problem in regression. In addition, two new measures are also proposed in this paper for measuring performance for imbalance multi-class classification.

A. MOTIVATIONAL EXAMPLES

Let us demonstrate some motivating examples for our work in real-life scenarios. Like binary classification, in multi-class classification, minority classes hold more importance than the majority classes. Let us consider a medical diagnosis centre. Suppose, there are three types of diseases: Flu, Hepatitis B and Cancer. Here, naturally the number of Cancer patients will be much less than Flu and Hepatitis B patients. Again, the number of Hepatitis B patients will be less than the number of Flu patients. Both Hepatitis B and Cancer are the minority classes and naturally more important to classify properly. Another real-life example for multi-class imbalance problem is oil bearing recognition. There are four classes of layers: oil layer, inferior oil layer, water layer and dry layer. Dry layer outnumbers the other layers, that is the other three layers are minority classes. But misclassifying oil layer to other layers is more costly. From the given examples, it should be clear why classifying minority classes are more important in some situations. However, as the majority classes outnumber the minority classes, the classifying model may get biased when classifying the minority classes.

As said before, data imbalance also occurs commonly in regression problems. Most real-world data distributions fall under the normal distribution which is bell shaped, and data that lies in the mid-range is often much higher in volume than that of the lower and upper ranges. For example, let us consider the grades of the students of a class. Statistics tell us that the majority of students are likely to receive an average grade, while the number of higher/lower grades will be significantly less. Interestingly, this scenario is also observed among diabetes patients too. Sugar level is a numeric value and the number of patients with very high or low sugar levels will be much lower than the number of patients with an acceptable sugar level. This also emphasizes the need to correctly classify minority classes. But, the training data will naturally hold more data of the middle ranges. The model may get biased towards the middle range values while predicting the lower or higher range values. Thus, it is vital to solve the data imbalance problem in regression analysis.

Most algorithms and ongoing research for solving the class imbalance problem is focused on binary classification, and does not scale well when the number of class grows. Many of them suffer from overfitting issues due to oversampling, and loss of critical information due to undersampling. Bagging and boosting techniques may still be penalized at each step, since they may still suffer from the class imbalance problem on the sampled data, when it has similar class distribution as in the original data. Again, the data imbalance problem also exists in regression analysis, although most algorithms do not consider this problem. This has been the driving force behind our proposal of effective method of solving the data imbalance problem.

B. CONTRIBUTIONS

The contributions of this paper are stated below:

- We propose an effective ensemble method for solving multi-class imbalanced problem.
- The data imbalance problem in regression analysis has been addressed.
- A new recursive approach is introduced for multi-class imbalance classification and regression for imbalanced data.
- We introduce new measures for calculating the recall and precision of multiple minority classes.
- Real-life applications of such approach indicate the suitability of the research work.
- Extensive experimental analyses prove the supremacy of the proposed approaches.

The remaining part of this paper is organized as follows. In Section 2, we present some of the most related works to get insight of our proposed work. An elaborate description and explanation of our proposed method with necessary terminologies, procedures, examples and pseudo-code is provided in Section 3. We have extensively analyzed our work experimentally to prove its supremacy and Section 4 is devoted for presenting those results. Finally, Section 5 concludes our research with some discussions and also some future scopes.

II. RELATED WORK

In this Section we will discuss some of the popular methods [12]–[16] for solving class imbalance problems. We shall discuss the methods proposed for multi-class classification problems [17]–[21]. We have also discussed some popular algorithms for regression.

A. RELATED WORK FOR IMBALANCE CLASSIFICATION

The techniques used to handle class imbalanced data can roughly be divided into two categories, External Methods or Data Level approaches and Internal Methods or Algorithmic Level approaches.

1) EXTERNAL METHODS

External methods primarily focus on shuffling and redistributing the training data in order to balance the classes. The external methods have the advantage of being able to apply any state of the art classification technique on the balanced data. External methods include Sampling, Bagging, SMOTE, etc. Since our proposed algorithm is external in nature, we focus more on the external methods.

There are two types of sampling techniques namely undersampling and oversampling. Oversampling works by resampling the tuples of the minority class so that the number of tuples from majority and minority classes in the training set become equal. Undersampling works by decreasing the number of tuples from the majority class [1].

Several variations of different sampling methods has been proposed and employed to deal with imbalanced classes. Japkowicz [9] primarily discussed two strategies: Undersampling and Resampling, asserting that both of these approaches are effective. They also observe and argue that using sophisticated sampling techniques for the class imbalance problem does not provide any advantage. Their findings are corroborated by Mani [10], who also states that complex undersampling techniques are outperformed by simple random undersampling strategies. Kubat [22] proposed a heuristic based undersampling method called One-Sided Selection which eliminates noisy or borderline majority classes. SMOTE [7] is a popular oversampling technique which generates synthetic instances of the minority class. The borderline-SMOTE [42] proposed by Han oversamples the minority class instances near the borderline. Xie *et al.* [43] found out that in general, oversampling performs better than undersampling. Sampling based methods change the original data class distribution of the imbalanced data which lead to some unexpected mistakes. For example, oversampling might lead to overfitting and undersampling may lose some useful information. Sampling methods can also cause dataset shift of the balanced data from the original imbalance data.

Bagging [23] is an ensemble method proposed by Breiman. Bagging generates new training sets each of the size of the original dataset by sampling from the dataset with replacement. Final classification result is obtained by majority votes [1].

Shuo Wang and Xin Yao combined Bagging techniques like UnderBagging, OverBagging and SMOTEBagging with special sampling methods [8]. In UnderBagging, each subset is created by undersampling majority classes randomly. In the similar way, OverBagging forms each subset by oversampling minority classes randomly. Majority vote is used for classifying a new instance. SMOTEBagging is different from UnderBagging and OverBagging because it generates synthetic instances for creating the subsets. For each iteration the bagging algorithms may still suffer from the class imbalance problem as the sampled subset in a given iteration has the similar class distribution with the original data set.

Author Zhongbin Sun devised two novel ensemble strategies for binary class imbalance problem, called SplitBal and ClusterBal [11]. The proposed methods are different from any other external methods, including Sampling and Bagging. This approach converts an imbalanced data set into several balanced ones in the first step. Then a number of classification techniques are applied on these multiple sets with a specific classification algorithm. The last step consists of combining all these classifiers into an ensemble one. Their method however only works for binary classes. Another drawback of their algorithm is that they use clustering as their data balancing method which cannot guarantee balance data and also puts the similar instances in one partition, so the partitions do not represent the original class, but a portion of that class.

2) INTERNAL METHODS

Instead of attempting to balance data, Internal or Algorithmic level approaches alters the existing algorithms to build their ability to learn from minority class [24], [25]. There are various commonly used methods such as cost sensitive and one-class learning.

Boosting [26] is a cost sensitive method where, weights are assigned to each training tuple. A series of classifiers are iteratively learned. After a classifier is learned, the weights are updated to allow the subsequent classifier to focus on the training tuples that were misclassified by the previous classifier. The final boosted classifier combines the votes of each individual classifier [1].

Like Bagging, Boosting based ensemble methods have also been popular for solving the class imbalance problem. Metacost [27] is a cost-sensitive learning method which uses bagging to find out optimal class labels for training data and then relabeling instances accordingly. Seiffert *et al.* [28] conducted a comprehensive study comparing sampling methods with boosting for improving the performance of decision trees model built for identifying the software defective modules. Their results showed that sampling methods were effective in improving the performance of such models while boosting outperformed even the best data sampling methods. Chawla [29] proposed a novel method called SMOTEBoost, which combined the SMOTE algorithm and some boosting algorithms to learn from imbalanced data sets. Seiffert *et al.* [30] presented a different hybrid ensemble methods named RUSBoost, which combined the random undersampling method and boosting. EasyEnsemble [31] is another method based on samping and boosting algorithms, which applies random undersampling methods on the majority class. EUSboost [32] also uses resampling, ensemble building, and voting for the final classification. PUSBE [33] is a novel ensemble algorithm where a filter method for feature selection is carried out. Another ensemble algorithm was proposed by Krawczyk *et al.* [34], which uses a cost-sensitive basic classifier internally, and also enables stochastic evolutionary algorithm to fusion basic classifier. Prati *et al.* [35] focused on setting up a series of experiments to assess the performance of some proposed treatment methods like SMOTE [7], ADASYN [36] and MetaCost [27] for imbalanced data. Ensemble methods based on boosting techniques often change the original data distribution. This is because they use sampling methods to balance the majority class instances or increase the minority class instances.

There are two popular approaches for multi-class classification problems [37]. They are: One-Versus-All (OVA) and All-Versus-All (AVA). In One-Versus-All, for n classes, n number of binary classifiers are built. A Classifier is trained using tuples of that class as the positive class, and the remaining tuples as the negative class. In One-Versus-All, given n classes, $\frac{n(n-1)}{2}$ binary classifiers are constructed. To classify an unknown tuple, each classifier votes and the class with the maximum number of votes wins. The problem with these

methods is that binary classifiers are sensitive to errors and those errors can affect the vote count [1].

B. RELATED WORK FOR REGRESSION

There are quite a few algorithms to address the regression problem [38], [41] but very few handles the imbalance problem. Linear Regression is the most basic form of regression, where for a single predicate value, the output variable is a linear function of the predicate value [1]. Another popular method is Classification and Regression Tree, also known as CART [39]. The representation for the CART model is a binary tree. Each root node represents a single input variable and a split point on that variable. The leaf nodes of the tree contain an output variable which is used to make a prediction.

An algorithm called Piecewise Regression or Segmented Regression has been proposed for the situation where independent variables, clustered into different groups, exhibit different relationships. Here, the independent variable is partitioned into intervals and a separate line segment is fit to each interval. The boundaries between the segments are breakpoints. Piecewise Linear Regression is a variation of Piecewise Regression where the relations in the intervals are obtained by Linear Regression. But these algorithms do not solve the data imbalance problem.

Locally Weighted Regression is another popular method for regression analysis. It performs a local regression around a point of interest, which only keeps all training data which are local with relative to the point of interest. Author Torgo [40] utilized sampling methods to address the imbalance problem in regression, however, such sampling methods cause problems like overfitting.

To overcome the limitations in these existing methods, we propose a new algorithm for imbalance classification of multiple classes and also extend it to regression for imbalance data. Moreover, our substantively new and distinct contributions beyond our preliminary conference version [2] include enhanced motivation which defines scope of real-life applications, enriched background study, revised algorithms with elaborate explanations, extensive experimental analyses on more real-life databases, t test for both imbalance classification and regression algorithm with respect to the existing methods which further establishes the efficiency of our methods.

III. PROPOSED ALGORITHM

Both of our algorithms are thoroughly described in this Section. Along with a comprehensive description of our algorithm, we have provided examples and pseudocodes for further clarity.

A. PROCEDURE

In this section, we will describe the step by step procedure of our algorithm for multi-class imbalance classification. We will also describe our algorithm for identifying and solving the data imbalance problem for regression.

1) METHOD FOR MULTI-CLASS IMBALANCE CLASSIFICATION

In our procedure for multi-class imbalance classification, the data imbalance problem is converted into multiple balanced problems. If necessary, this process of data balancing is repeated recursively and it is bounded by a certain threshold. An ensemble classifier is modeled and one class is selected with the help of an ensemble rule. The advantage of our proposed method over other methods (Underbagging, sampling etc) can be described in two folds. First, we use effective technique for balancing the imbalanced data. Second, we introduce unique recursive method that repeats the data balancing procedure multiple times for partitioning the imbalance data into multiple balanced data. The step by step description of our algorithm is given below:

1) **Data Partitioning:**
It is possible that there are multiple majority and minority classes in multi-class imbalance problems. However, in our method the class containing most number of records (r_{maj}) is referred as majority class(C_{maj}) and similarly the class containing least number of records (r_{min}) is referred as minority class(C_{min}). All the classes except the minority class(C_{min}) is partitioned using a data partitioning method. Each partition will contain r_{min} number of records. For a class C_x containing r_{cx} records, the total number of partitions will be r_{cx}/r_{min} if r_{cx} % $r_{min} = 0$, otherwise it will be $(r_{cx}/r_{min}) + 1$. We have used two data partitioning methods called Partition using Balanced Distribution(PBD) and Random Partitioning(RP).

   - **Partition using Balanced Distribution**
   The main purpose of this method is to keep the data distribution in the partitions same as the data distribution in the original class. Suppose the number of records in the minority class C_{min} is r_{min} and a class C_x which is not the minority class has r_{cx} number of records. For partitioning class C_x using balance distribution, first r_{cx}/r_{min} number of partitions are created. A random record is selected and its $(r_{cx}/r_{min}) - 1$ nearest records are selected. So in total we have $1 + (r_{cx}/r_{min} - 1) = r_{cx}/r_{min}$ records. They are assigned to the r_{cx}/r_{min} partitions (one record per each partition). This whole process is repeated r_{cx} number of times. This ensures that each partition represents the whole class other than representing a part of that class only because each of the partitions holds all types of data of the class. After finishing the whole process, if any records are left then a new partition is created and they are assigned to that partition.
   - **Random Partitioning**
   In case of this partitioning method, all the classes except the minority class are randomly partitioned into r_{cx}/r_{min} or $(r_{cx}/r_{min}) + 1$ partitions.

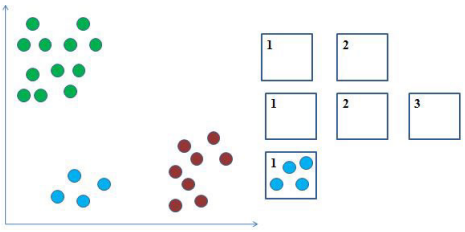

FIGURE 1. Multi-class imbalance dataset.

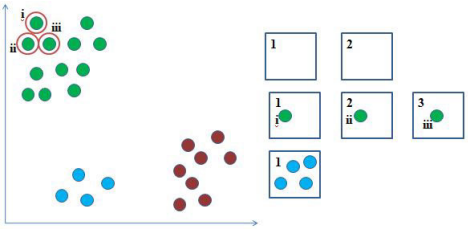

FIGURE 2. Partitioning using balanced distribution.

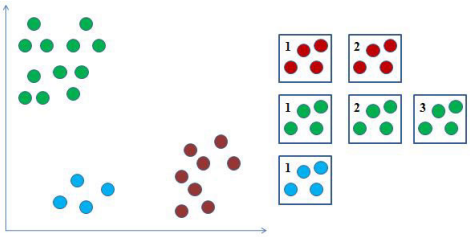

FIGURE 3. Partitions of the classes.

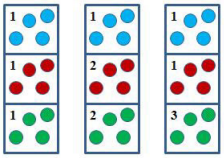

FIGURE 4. Balanced data bins.

2) **Balanced Data Creation:**
The data partitions that are created in the previous step are used for generating multiple balanced data bins. The total number of balanced data bins is equal to the number of partitions of the majority class. First, the data partitions of the majority class are assigned to the bins, one partition per each bin. After that, the data partitions of the other classes are sequentially assigned to the balanced data bins and repeated if necessary.
3) **Building Ensemble Classifier:**
Each of the balanced data bins are used for building a classifier. All these classifiers are used for constructing an ensemble classifier.
4) **Classifying Test Data**
For a new test data, a class label is assigned using the ensemble classifier. The popular ensemble rule Majority Voting is used as our ensemble rule. In this rule, a vote is assigned to a class if a classifier selects that class and the class that is assigned the most number of votes wins.
5) **Recursive Approach**
In the first step of our method, as the rest of the r_{cx} % r_{min} records are assigned to a new partition, so class imbalance problem may still exist in these data bins. In that case, the bins are again partitioned into balanced bins by recursive call. This unique recursive approach sets our algorithm apart from the traditional ones and makes the process much more effective.

Let us demonstrate our proposed method with an example. In Figure 1, we can see three classes indicated by three different colors. Clearly class imbalance problem exists in the data. The Green ones represent the majority class (12 data instances) and Blue represents the minority class (4 data instances). We do not need to partition the Blue class as it is the minority class. However, we have to partition the Red and Green classes. For the Green class, number of partition, $N_{partition} = 12 \div 4 = 3$, as $12\%4 = 0$. For Partition using Balanced Distribution, each time we will randomly select one data instance and select it's (3-1) = 2 nearest neighbours. On the other hand for the Red class, number of partition, $N_{partition} = 8 \div 4 = 2$, as $8\%4 = 0$. Each time we will select (2-1) = 1 nearest neighbors for a randomly selected data instance. Suppose, for the Green class, we randomly select the data instance indicated with *i* in Figure 2. Then we select its two nearest neighbors (indicated by *ii* and *iii*). These three data instances are assigned to the three different data partitions, one at each. This will continue until all the three partition size is equal to the size of the minority class, that is the Blue class. Following the same procedures, the Red class is divided into two partitions. In Figure 3, we can see the partitions of all the classes.

In the next step we create a number of balanced data bins using the partitions. The number of balanced data bins is equal to the number of partitions of the majority class. Here, the total number of balanced data bins will be 3 as the majority class Blue has 3 partitions. We take the 3 partitions of the majority classes and assign them to the balanced data bins, one at each time. Then we assign the partitions of the other classes sequentially and repeat if necessary. In Figure 4, we assign the partitions of the Green class to the balanced data bins, one at each. Then we assign the partitions of the Red class (partition 1 and 2) and then repeat partition 1 again. Similarly we assign the only partition of the Blue class to all the bins. Finally we get three balanced data bins. Next, each of the balanced data bins are used for building a classifier. All these classifiers are used for constructing an ensemble classifier. In Figure 5 we can see that this ensemble classifier is used for classifying new data using an ensemble rule.

2) METHOD FOR SOLVING DATA IMBALANCE PROBLEM FOR REGRESSION

We have introduced an effective recursive approach for solving the data imbalance problem of regression by extending our algorithm for data imbalance multi-class classification. Our proposed algorithm is described below:

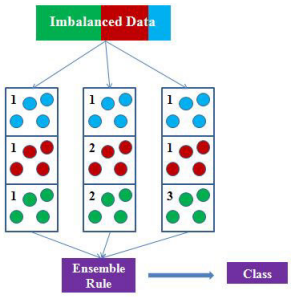

FIGURE 5. Building ensemble classifier and classifying new data using ensemble rules.

1) **Data Imbalance Identification:**
The data imbalance problem in regression analysis is identified by dividing the range (smallest value to largest value) of values into smaller ranges of equal size and calculating the records of each smaller range. At first, the range is divided into three equal size ranges. If the ratio of the number of records of any two smaller ranges is greater than or equal to 1.5, then it is identified that data imbalance problem exists in the data. In that case the data belonging to each range is assigned a group. For the next steps of the algorithm, these groups are treated as class labels of the data. If the ratio is smaller than 1.5 than the number of smaller range is increased by one and again imbalance is checked in the same process. For example, if we cannot find data imbalance by dividing the range into three equal sized ranges, then next step is to divide it into four equal sized ranges. It is repeated till a user specified threshold is reached. The threshold for imbalance ratio is set to 1.5 as the popular data repositories like Keel identifies a dataset as imbalanced if the imbalance ratio between any two class is more than or equal 1.5.
2) **Modeling Ensemble Classifier**
This step is the similar to the steps from 1 to 4 of our multi-class imbalance data classification algorithm. The groups found from the previous step are treated as their class. At the end of this step a group/class label is found. The steps are thoroughly described below:
   - **Data Partitioning:**
   The group containing most number of records ($Group_{maj}$) is referred as majority class and similarly the group containing least number of records ($Group_{min}$) is referred as minority class. All the groups except the minority group is partitioned using the data partitioning method using balanced distribution described in our previous algorithm. Each partition will contain $Group_{min}$ number of records. For a group $Group_x$ containing $Group_{cx}$ records, the total number of partitions will be $Group_{cx}/Group_{min}$ if $Group_{cx}$ % $Group_{min} = 0$, otherwise it will be $(Group_{cx}/Group_{min}) + 1$.
   For partitioning group $Group_x$, which is not a minority group, first we will partition it into $Group_{cx}/Group_{min}$ partitions. A random record is selected and its $(Group_{cx}/Group_{min}) - 1$ nearest records are selected. So in total we have $1 + (Group_{cx}/Group_{min} - 1) = Group_{cx}/Group_{min}$ records. They are assigned to the $Group_{cx}/Group_{min}$ partitions (one record per each partition). This whole process is repeated $Group_{cx}$ number of times. This ensures that each partition represents the whole group other than representing a part of that group only because each of the partitions holds all types of data of the group. After finishing the whole process, if any records are left then a new partition is created and they are assigned to that partition.
   - **Balanced Data Creation:**
   The data partitions that are created from the previous step are used for generating multiple balanced data bins. The total number of balanced data bins is equal to the number of partitions of the majority group. First, the data partitions of the majority group are assigned to the bins, one partition per each bin. After that, the data partitions of the other groups are sequentially assigned to the balanced data bins and repeated if necessary.
   - **Building Ensemble Classifier:**
   Each of the balanced data bins are used for building a classifier. The group labels of the data are used as their class label. All these classification models are used for constructing an ensemble classifier.
   - **Classifying Test Data**
   For a new test data, a class or group label is assigned using the ensemble classifier. The popular ensemble rule Majority Voting is used as our ensemble rule.
3) **Value Prediction**
A prediction value is acquired by applying an algorithm for regression analysis on the data of the selected group/class from the previous step. Results show that by narrowing down the search space like this, higher efficacy is achieved.
4) **Recursive Technique**
As the predicted value is continuous in case of regression analysis, the data in the selected group can again suffer from data imbalance problem. For solving this problem, the same process is repeated on the data of the selected group. This recursive process is repeated for a specific threshold. This recursive based approach makes our method unique and remarkably increases the effectiveness of the algorithm.

Let us see our proposed method with an example. Figure 6 shows a regression dataset where the predicted values are people's ages. The range of the predicted value is 1 to 60. So we divide the range into three equal smaller ranges. After that, we count the number of data instances for each range. Here, the ratio of the smallest and largest number of data is $(9 \div 3) = 3$. As it is greater than 1.5, so it is identified

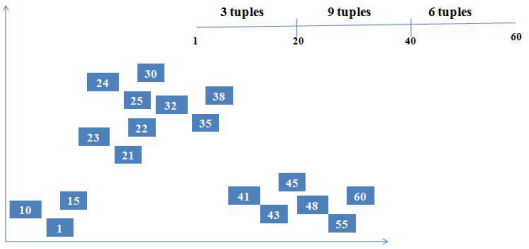

FIGURE 6. Identifying data imbalance problem in regression problem.

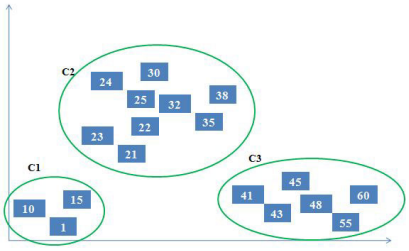

FIGURE 7. Partitioning the data for regression into groups.

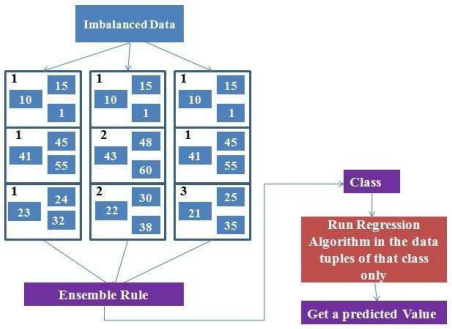

FIGURE 8. Balanced regression.

that data imbalance problem exists in the data. Now, each range of data instances is given a group or class label which is shown in Figure 7.

Next, the groups or classes are partitioned so that the size of each partition is equal to the size of the minority class or group. These partitions are used for creating a number of balanced data bins. In Figure 8, an ensemble classifier is constructed with these balanced data bins. The ensemble classifier assigns a class or group label to a new data. But since this is a regression problem, we need to predict a numeric value. So a standard regression algorithm is applied, but only on the data instances of the group or class that is selected by the ensemble classifier. For example, for a new test data, if the ensemble classifier selects the class $C3$, then a standard regression algorithm will be applied on the data instances of $C3$ only. As mentioned earlier, data imbalance problem may still occur in the selected group. For solving this problem, process of selecting the group is repeated recursively bounded by a user specified threshold. In that case $C3$ will be further divided into a number of smaller ranges and if there is data imbalance problem, then the whole process will be repeated again. In our experiments we have set the threshold to two.

B. PSEUDOCODE

In this section we will present the pseudo-code of our proposed algorithm. Algorithm 1-4 represent our algorithm for multi-class imbalance problem. For partitioning method, Partition using Balanced Distribution is used. Algorithm 5-7 represent our algorithm for identifying and solving regression with data imbalance problem.

Algorithm 1 ImbalanceMulticlassClassification

1 $Balance_{num} = N_{maj}/N_{min}$
2 **for** *each class* C_i *where* $i = 1,2,...n$ **do**
3 $\quad Partition_{Ci} \leftarrow Partition(C_i, N_{min})$
4 **end**
5 $BalancedBin =$ $CreateBalanceBin$(CreateBalanceBin($Balance_{num}$, $Partition_{C_{1,2,...n}}$)
6 set *RecursionLimit*
7 *BalanceClassify*(*BalancedBin*, 1, *RecursionLimit*)

In Algorithm 1, in the first Line, the number of balanced bin is calculated. From Lines 2-3 each class in partitioned into equal number of partitions. In Lines 4-5, these partitions are used for creating a number of balanced data bins. In Line 6, a *RecursionLimit* is set which determines how many times the data balancing process will be repeated if data is not balanced. In Line 7, new data are classified by calling the method *BalanceClassify* and the *RecursionLimit* is sent as a parameter.

In algorithm 2, from Line 1-6, a class is partitioned into a number of partitions. In Line 7-9, the partitions are assigned to a set. From Line 10-13, a new partition is assigned to the set with the data that were left behind in Line 1-6. In Line 14, the set containing the partitions is returned.

Algorithm 2 Partition(C_i, N_{min})

1 **for** $k = 1,2,...N_{min}$ **do**
2 $\quad e_o \leftarrow$ randomly selected record of C_i
3 $\quad e_j \leftarrow jth$ nearest record of e_o, where $j = 1, 2, ...N_{C_i}/N_{min}$
4 $\quad Partition_{ci}.j \leftarrow Partition_{ci}.j \cup e_j$
5 $\quad C_i = C_i \setminus e_j$, where $j = 1, 2, ...C_i/N_{min}$
6 **end**
7 **for** *each* $Partition_{ci}.j$ *where* $j = 1, 2, ...N_{C_i}/N_{min}$ **do**
8 $\quad Partition_{ci} \leftarrow Partition_{ci}.j$
9 **end**
10 **if** *class* C_i *is non-empty* **then**
11 $\quad$ Create another partition $Partition_{ci}.j++$ with the records of C_i
12 $\quad Partition_{ci} \leftarrow Partition_{ci}.j$
13 **end**
14 return $Partition_{ci}$

Algorithm 3 CreateBalanceBin($Balance_{num}$, $Partition_{C_{1,2,...n}}$)

```
1  for each balanced bin B_i where i = 1, 2, ....Balance_num do
2  |  set B_i as empty
3  end
4  for each class C_i where i = 1,2,....n do
5  |  for each balanced bin B_j where j = 1,2,....Balance_num do
6  |  |  k = (i-1)% Size of class C_i + 1
7  |  |  B_j ← Partition_ci.k
8  |  end
9  end
10 for each balanced bin B_i where i = 1,2,....Balance_num do
11 |  BalancedBin ← B_i
12 end
13 return BalancedBin
```

In Algorithm 3, Line 1-3, all the balanced bins are initially set as empty. From Line 4-9 a number of balanced bins are created by sequentially adding partitions of classes in the balanced bins and repeating if necessary. In Line 10-12, these balanced bins are assigned to a set called *BalancedBin*. The set is returned in Line 13.

In Algorithm 4, new test data are classified using the ensemble classifier. A recursion limit is set, which decides how many times the data balancing process will take place if there exists imbalance is the balanced bins. In Lines 1-8, if the threshold is exceeded or the data bins are balanced, then the new test data are classified using the ensemble classifier which uses a particular ensemble rules. The threshold for imbalance ratio is set to 1.5 in accordance with popular data repositories like Keel. Else, in Lines 10-14, the imbalanced bin is balanced by calling Algorithm 2 and 3. Then from Lines 15-17 the data are classified by recursive call of the Algorithm.

Algorithm 4 BalanceClassify(*BalancedBin*, *Threshold*, *RecursionLimit*)

```
1  if threshold > recursion_limit or ImbalabceRatio(BalancedBin)≤ 1.5 then
2  |  for each B_i in BalancedBin where i = 1,2,...Balance_num do
3  |  |  Train B_i using a classification model
4  |  end
5  |  for each test data do
6  |  |  Assign class label according to an ensemble rule
   |  |  using the result of the ensemble classifiers.
7  |  end
8  end
9  else
10 |  for each Partition_Ci.BalancedBin where i = 1,2,...n do
11 |  |  Partition = Partition(Partition_Ci.BalancedBin,
   |  |  min(Partition_Ci.BalancedBin))
12 |  end
13 |  BalancedData ← CreateBalanceBin(Balance_num,
   |  Partition)
14 end
15 for each balanced bin b_i in BalancedData do
16 |  BalanceClassify(b_i, Threshold + 1 )
17 end
```

In Algorithm 5, from Line 1-6 the data imbalance problem is identified and the data is divided into groups accordingly. In Line 7, the number of balanced bin is calculated. From Lines 8-10, each class in partitioned into equal number of partitions. In Line 11 these partitions are used for creating a number of balanced data bins by using Algorithm 3, where the groups are treated as classes. In Line 12, for *BalanceRegResult* a recursive limit is set. In Line 13, new data are predicted.

Algorithm 5 BalanceRegression

```
1  Set GroupSize
2  Set ImbalanceThreshold
3  CreateGroup(GroupSize)
4  while Group_maj/Group_min < ImbalanceThreshold do
5  |  CreateGroup(GroupSize + 1)
6  end
7  Balance_num = Group_maj/Group_min
8  for each group G_i where i = 1,2,...n do
9  |  Partition_Gi ← PartitionReg(G_i, Group_min)
10 end
11 BalancedBin = CreateBalanceBin(Balance_num,
   Partition_Group_1,2,...n)
12 set RecursionLimit
13 BalanceRegResult(BalancedBin, 1, RecursionLimit)
```

In algorithm 6, from Line 1-6, a group is partitioned into a number of partitions. In Line 7-9, the partitions are assigned to a set. From Line 10-13, a new partition is assigned to the set with the data that were left behind in Line 1-6. In Line 14, the set containing the partitions is returned. These partitions are used for creating a number of balanced data bins using algorithm 3, where the groups are treated as classes.

In Algorithm 7, new test data are predicted using the ensemble classifier. A recursion limit is set, which decides how many times the process will take place if there exists imbalance is the balanced bins. In Lines 1-9, if the threshold is exceeded, then the new test data are predicted using the ensemble classifier which uses a particular ensemble rules and selects a group/class. The final predicted value is found by applying an algorithm for regression analysis on the data of the selected group only. Otherwise, if the condition in Line 10-13 is not met, then the whole process is again repeated on the data of the selected group by calling Algorithm 5 and the value of the threshold is incremented.

Algorithm 6 PartitionReg(G_i, $Group_{min}$)

for $k = 1,2,...Group_{min}$ **do**
  $e_o \leftarrow$ randomly selected record of G_i
  $e_j \leftarrow jth$ nearest record of e_o, where $j = 1,2,....G_i/Group_{min}$
  $Partition_{gi}.j \leftarrow Partition_{gi}.j \cup e_j$
  $G_i = e_j \setminus e_j$, where $j = 1,2,....G_i/Group_{min}$
end
for *each* $Partition_{gi}.j$ *where* $j = 1,2,....G_i/Group_{min}$ **do**
  $Partition_{gi} \leftarrow Partition_{gi}.j$
end
if *group* G_i *is non-empty* **then**
  Create another partition $Partition_{gi}.j++$ with the records of G_i
  $Partition_{gi} \leftarrow Partition_{gi}.j$
end
return $Partition_{gi}$

Algorithm 7 BalanceRegResult(*BalancedBin*, *Threshold*, *RecursionLimit*)

if $threshold > recursion_{limit}$ **then**
  for *each* B_i *in BalancedBin where* $i = 1,2,...Balance_{num}$ **do**
    Train B_i using a classification model using the group label as class label
  end
  for *each test data* **do**
    Assign group/class label according to an ensemble rule using the result of the ensemble classifiers.
    Predict the value by using a standard algorithm for regression only on the data of the assigns group/class.
  end
end
else
  BalanceRegression(b_i)
  $Threshold \leftarrow Threshold + 1$
end

C. OUR PROPOSED PERFORMANCE MEASURES

Apart from using Average Recall, Average Precision and F measure, we have also proposed two new measures for evaluating performance for multi-class classification problem. As we have discussed earlier that the minority classes are more important than the majority classes. However, in multi-class classification problems, there are multiple minority classes. For analyzing the performances of the minority classes, their individual results are generated. But there is no such measure which can represent the results of the minority classes at a time. For this reason, we have proposed two measures called *Min Average Recall* and *Min Average Precision* which represent the results of all the minority classes. They are discussed below:

- **Min Average Recall:** Min Average Recall is defined as the average of the recalls of the minority classes. If the *Ratio* of the majority class and a class is less than or equal to a specified threshold threshold, than that class is considered a minority class.
- **Min Average Precision:** Min Average Precision is defined as the average of the precisions of the minority classes. Here also, if the *Ratio* of the majority class and a class is less than or equal to a specified threshold, than that class is considered a minority class.

In both the cases the threshold is set to 1.5 just like other standard dataset respiratories.

IV. EXPERIMENTAL RESULTS

In this Section we have presented the experimental results achieved by using our proposed methods. We have conducted a comprehensive performance study on our experiments that were run on real world datasets with various performance measures. The results indicate that our proposed algorithms are effective for multi-class imbalance classification and data imbalance regression problem. We have compared the performancce of our multi-class imbalance classification algorithm with the performances of two other algorithms (SMOTEBagging and SplitBal). We also have compared the performance of our algorithm for data imbalanced regression with the performances of four algorithms (Linear Regression, REPTree, Locally Weighted Regression, Piecewise Regression). The comparison of performance has shown that our algorithms are more effective.

All the experiments of both our methods and the other methods are implemented and run on PC with Intel Core i5 processor, 4 GB RAM capacity and Windows 7 as operating system. Waikato Environment for Knowledge Analysis (known as WEKA) software is used for implementing the base classifiers and regression models and we used JAVA programming language for implementing all the algorithms.

A. PERFORMANCE ANALYSIS FOR MULTI-CLASS IMBALANCE CLASSIFICATION

For evaluating the performance of our proposed algorithm for multi-class imbalance classification, we have performed experiments on 10 different real life datasets. They all were collected from Keel Dataset Respiratory. All of them have multiple classes with imbalance problem. The datasets along with their class number and Imbalance Ratio(IR) are: Balance (Total number of classes: 3, IR: 5.88), Dermatology (Total number of classes: 6, IR: 5.55), New Thyroid (Total number of classes: 3, IR: 4.84), Wine (Total number of classes: 3, IR: 1.5), Splice (Total number of classes: 3, Imbalance Ration: 2.16), Led7digit (Total number of classes: 10, Imbalance Ration: 1.54), Zoo (Total number of classes: 7, IR: 10.25), Hayes Roth (Total number of classes: 3, Imbalance Ration: 1.7), Car (Total number of classes: 4, Imbalance Ration: 18.62), Satimage (Total number of classes: 7, Imbalance Ration: 2.45). The performance measures that we used for evaluating the performances of the methods are Average

TABLE 1. Average recall results.

Method	Dataset	Naive Bayes	j48	IBK	jrip	oneR	R. Subspace	R. Forest
PBD	Dermatology	95.78	97.17	97.5	97.83	97.22	97.89	97.43
	New Thyroid	95.41	95.48	95.26	96.48	96.74	96.89	97.26
	Splice	95.14	95.20	95.01	95.26	95.29	95.23	95.12
	Zoo	90	89.05	88.70	89.10	88.50	88.12	89.29
	Wine	97.87	97.59	97.69	97.5	97.5	97.45	97.96
RP	Dermatology	97.28	97.17	97.22	97.33	97.67	97.83	96.89
	New Thyroid	95.33	95.19	94.52	94.89	95.11	95.63	95.11
	Splice	95.30	95.24	95.23	95.24	95.16	95.28	95.29
	Zoo	87.13	87.24	87.99	88.71	89.12	88.83	88.62
	Wine	97.87	98.15	97.83	97.82	97.82	97.96	98.06

TABLE 2. Min average recall results.

Method	Dataset	Naive Bayes	j48	IBK	jrip	oneR	R. Subspace	R. Forest
PBD	Dermatology	95	96.33	97.07	97.47	96.73	97.53	96.73
	New Thyroid	94.78	94.67	94.44	94.22	94.56	94.11	94.22
	Splice	95.17	95.03	95.30	95.36	95.48	95.37	95.30
	Zoo	88.33	86.94	87.82	88.42	87.63	87.07	88.22
	Wine	100	100	100	100	100	100	100
RP	Dermatology	97.4	96.73	96.67	96.8	97.4	97.4	96.33
	New Thyroid	94.56	94.44	94	94.22	94.11	94.56	94
	Splice	95.23	95.12	95.36	95.38	95.42	95.41	95.49
	Zoo	85.90	86.63	87.36	88.17	87.78	88.03	87.11
	Wine	100	100	100	100	100	100	100

Recall (average of the Recalls of all the classes), Min Average Recall, Average Precision (average of the Precisions of all the classes), Min Average Precision and F measure. Among them, Min Average Recall and Min Average Precision are proposed by us and they are described in Section 3. We have used cross−validation for all the experiments and majority voting is used as the ensemble rule for the ensemble classifier.

1) AVERAGE RECALL

In this section we have presented the experimental results for Average Recall on five real life datasets with class imbalance problem with multiple classes. We have performed experiments for both Partition using Balanced Distribution (PBD) and Random Partitioning (RP). For base classifiers we have used seven popular classification algorithms: Naive Bayes, j48, KNN, Ripper, OneR, Random Forest and Random Subspace. In all the experiments the number of test data for each class is the same. Table 1 shows the results of Average Recall for Partitioning using Balanced Distribution and Random Partitioning for various Datasets.

2) MIN AVERAGE RECALL

In this section we have presented the experimental results for Min Average Recall on five real life datasets with class imbalance problem with multiple classes. In this case also, we have performed experiments for both PBD and RP. Here also, for base classifiers we have used seven popular classification algorithms: Naive Bayes, j48, KNN, Ripper, OneR, Random Forest and Random Subspace. In all the experiments the number of test data for each class is the same. Table 2 shows the results of Min Average Recall for Partitioning using Balanced Distribution and Random Partitioning for various Datasets.

3) AVERAGE PRECISION

In this section we have presented the experimental results for Average Precision on five real life datasets with class imbalance problem with multiple classes. Here also, we have performed experiments for both Partition using Balanced Distribution (PBD) and Random Partitioning (RP). Like before, for base classifiers we have used seven popular classification algorithms: Naive Bayes, j48, KNN, Ripper, OneR, Random Forest and Random Subspace. In all the experiments the number of test data for each class is the same. Table 3 shows the results of Average Precision for Partitioning using Balanced Distribution and Random Partitioning for various Datasets.

4) MIN AVERAGE PRECISION

In this section we have presented the experimental results for Min Average Precision on five real life datasets with class imbalance problem with multiple classes. Like before, we have performed experiments for both PBD and RP. Here also, for base classifiers we have used seven popular classification algorithms: Naive Bayes, j48, KNN, Ripper, OneR, Random Forest and Random Subspace. Table 4 shows the results of Min Average Precision for Partitioning using Balanced Distribution and Random Partitioning for various Datasets.

5) F MEASURE

In this section we have presented the experimental results for F measure on five real life datasets with class imbalance problem with multiple classes. We have performed experiments for both Partition using Balanced Distribution (PBD) and Random Partitioning (RP). For base classifiers we have used seven popular classification algorithms: Naive Bayes, j48, KNN, Ripper, OneR, Random Forest and Random Subspace. In all the experiments the number of test data for each class

TABLE 3. Average precision results.

Method	Dataset	Naive Bayes	j48	IBK	jrip	oneR	R. Subspace	R. Forest
PBD	Dermatology	96.47	97.40	97.93	98.12	97.39	97.56	98.15
	New Thyroid	95.92	95.91	95.85	96.02	95.64	95.18	95
	Splice	95.18	95.18	95.05	95.29	95.33	95.27	95.17
	Zoo	92.73	90.74	91.92	92.03	92.23	92.07	91.69
	Wine	98.08	97.08	97.92	97.82	98.06	97.79	97.70
RP	Dermatology	97.69	97.56	97.61	97.71	97.94	97.96	98.04
	New Thyroid	95.92	95.72	95.05	96.02	95.64	95.18	95
	Splice	95.18	95.24	95.26	95.28	95.20	95.32	95.33
	Zoo	90.37	90.74	91.92	92.04	92.23	92.07	91.69
	Wine	98.10	98.13	98.1	98.05	97.88	98.08	98.19

TABLE 4. Min average precision results.

Method	Dataset	Naive Bayes	j48	IBK	jrip	oneR	R. Subspace	R. Forest
PBD	Dermatology	97.29	96.79	97.52	97.80	95.91	97.08	97.78
	New Thyroid	98.46	98.44	98.56	99.10	98.62	97.75	98.35
	Splice	95.11	95.21	94.73	95.06	95.07	95.02	94.86
	Zoo	91.52	90.12	90.32	90.73	90.50	89.82	91.06
	Wine	97.43	97.68	97.80	97.51	97.95	97.68	96.96
RP	Dermatology	97.29	97.08	97.13	97.26	97.53	97.67	97.72
	New Thyroid	98.57	98.44	97.90	98.32	98.88	98.63	98.97
	Splice	95.34	95.26	94.95	94.98	94.87	95.06	95.02
	Zoo	88.77	89.73	90.57	90.71	90.94	90.75	90.31
	Wine	97.70	97.68	97.46	97.78	97.60	97.46	97.83

TABLE 5. F measure results.

Method	Dataset	Naive Bayes	j48	IBK	jrip	oneR	R. Subspace	R. Forest
PBD	Dermatology	96.12	97.14	97.71	97.97	97.30	97.72	97.79
	New Thyroid	95.66	95.69	95.55	95.75	95.19	95.03	95.37
	Splice	95.16	95.22	95.03	95.27	95.31	95.25	95.14
	Zoo	91.34	89.53	90.18	90.56	90.15	89.67	90.79
	Wine	97.97	97.73	97.89	97.66	97.78	97.62	97.83
RP	Dermatology	97.48	97.36	97.41	97.52	97.80	97.89	97.46
	New Thyroid	95.61	95.45	94.78	95.21	95.44	95.61	95.64
	Splice	95.32	95.24	95.24	95.26	95.18	95.30	95.31
	Zoo	88.72	88.96	90.16	90.34	90.65	90.42	90.13
	Wine	97.98	98.08	98.46	97.93	97.85	98.02	98.12

is the same. Table 5 shows the results of F measure for Partitioning using Balanced Distribution and Random Partitioning for various Datasets.

6) COMPARISON WITH EXISTING ALGORITHMS

In this section we will compare our algorithm with two other external algorithms for class imbalance problem. They are: SMOTEBagging and SplitBal. SMOTEBagging is designed for multi-class imbalance problem. However SplitBal is designed for binary class imbalance problems only. For that reason we have used the AVA (All Versus All) method for making it applicable for multi-class problems as well. We have used both of our data balancing techniques: PBD (Partition using Balanced Distribution) and RP (Random Partitioning). Our algorithm provides better results for solving the multi-class imbalance problem.

In Table 6 we present the comparison results for Average Recall. The base classifier is j48. We have compared the results with both of our techniques: Partition using Balanced Distribution(PDB) and Random Partitioning(RP). The results show that our algorithms perform better than the other two algorithms as our recursive based method effectively partitions the majority classes which prevents data loss or overfitting.

TABLE 6. Average recall comparison(base classifier: J48).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	74.89	**75.17**	61.56	69.8
Dermatology	96.89	**97.17**	96.78	95.03
New thyroid	**95.48**	95.19	92.67	93.09
Wine	97.59	**98.15**	93.33	95.47
Splice	95.20	**95.24**	92.13	92.12
Led7digit	**71.76**	71.52	70.59	66.23
Zoo	**89.05**	87.99	85.89	88.19
Hayes Roth	**86.35**	86.33	84.22	84.19
Car	98.03	**98.98**	97.02	83.03
Satimage	**78.88**	78.45	75.23	75.25

In Table 7 we present the comparison results for Average Recall. The base classifier is Naive Bayes. Here also, we have compared the results with both of our techniques: Partition using Balanced Distribution(PDB) and Random Partitioning(RP). It shows that our methods perform better than the other two algorithms as our methods use effective data balancing technique and recursive based approach.

Table 8 shows the comparison results for Min Average Recall. The base classifier is j48. We have compared the results with both of our techniques: Partition using Balanced Distribution(PDB) and Random Partitioning(RP). It shows that our methods perform better than the other two algorithms

TABLE 7. Average recall comparison(base classifier: Naive Bayes).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	74.44	**75.83**	74	70.68
Dermatology	95.78	**97.28**	93.22	94.37
New thyroid	**95.41**	95.33	92.67	92.53
Wine	**97.87**	**97.87**	97.78	95.69
Splice	95.14	**95.30**	92.97	92.17
Led7digit	71.83	**72.03**	68.49	66.41
Zoo	**90**	87.13	89.11	88.24
Hayes Roth	**76.29**	74.89	73.11	73.55
Car	89.06	**89.17**	87.02	83.19
Satimage	**78.02**	76.99	75.01	75.26

TABLE 8. Min average recall comparison(base classifier: J48).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	**91**	80.67	14	68.9
Dermatology	96.33	**96.73**	96.27	94.44
New thyroid	**94.67**	94.44	92	92.53
Wine	**100**	**100**	94.17	97.58
Splice	95.03	**95.12**	93	91.97
Led7digit	**79.64**	78.06	76.03	73.71
Zoo	**86.94**	86.63	86.66	82.78
Hayes Roth	99.05	**99.09**	99.00	98.27
Car	98.05	**98.93**	98.00	94.26
Satimage	**66.99**	66.98	65.57	63.51

TABLE 9. Min average recall comparison(base classifier: Naive Bayes).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	78.67	**85.67**	73.33	67.85
Dermatology	95	**97.4**	91.87	94.12
New thyroid	**94.78**	94.56	94	92.2
Wine	**100**	**100**	**100**	98.27
Splice	95.17	**95.23**	90.41	92.17
Led7digit	88.43	**85.90**	75	72.18
Zoo	**88.33**	85.90	87.71	86.63
Hayes Roth	**100**	99.78	99.07	99.11
Car	98.09	**98.99**	98.02	93.08
Satimage	**66.88**	66.67	65.66	63.52

TABLE 10. Average precision comparison(base classifier: J48).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	**82.69**	79.98	55.63	73.85
Dermatology	97.40	**97.56**	97.09	95.39
New thyroid	**95.91**	95.72	93.53	93.04
Wine	98.08	**98.13**	94.03	95.94
Splice	**95.25**	95.24	93.16	92.25
Led7digit	**73.30**	73.15	71.86	67.51
Zoo	**91.34**	92.74	88.45	91.27
Hayes Roth	**85.65**	85.55	84.78	84.99
Car	97.27	**97.99**	97.19	84.46
Satimage	**77.10**	77.01	76.22	75.46

TABLE 11. Average precision comparison(base classifier: Naive Bayes).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	77.40	**81.50**	77.16	73.85
Dermatology	96.47	**97.69**	94.63	94.92
New thyroid	95.92	95.89	**96.40**	93.29
Wine	98.08	**98.10**	98.02	96.11
Splice	95.18	**95.34**	93.16	92.30
Led7digit	73.16	**73.54**	70.14	67.63
Zoo	**92.73**	90.37	89.35	91.48
Hayes Roth	**76.45**	74.75	72.77	72.99
Car	**89.75**	88.95	88.05	84.63
Satimage	**76.98**	76.94	75.95	75.48

TABLE 12. Min average precision comparison(base classifier: J48).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	58.31	**60.05**	33.46	56.09
Dermatology	96.79	**97.08**	96.64	94.92
New thyroid	**98.68**	98.44	97.11	96.21
Wine	97.68	97.68	**98.43**	97.59
Splice	95.21	**95.26**	90.25	90.71
Led7digit	78.18	**79.16**	74.23	68.90
Zoo	**90.12**	89.73	87.64	90.04
Hayes Roth	**99.92**	95.76	99.90	90.87
Car	**99.03**	99.01	98.26	88.19
Satimage	**66.76**	66.15	65.66	64.12

as our recursive based methods efficiently partitions the majority classes which prevents data loss or overfitting.

In Table 9 we present the comparison results for Average Recall. The base classifier is Naive Bayes. We have compared the results with both of our techniques: Partition using Balanced Distribution(PDB) and Random Partitioning(RP). Here also it shows that our methods perform better than the other two algorithms as our algorithms use effective data balancing technique and recursive based approach.

In Table 10 we present the comparison results for Average Precision. The base classifier is j48. We have compared the results with both of our techniques: Partition using Balanced Distribution(PDB) and Random Partitioning(RP). It shows that our proposed methods perform better than the other two algorithms as our recursive based methods effectively partitions the majority classes so that there is no loss of data or overfitting.

In Table 11 we present the comparison results for Average Precision. The base classifier is Naive Bayes. Here also, we have compared the results with both of our techniques: Partition using Balanced Distribution(PDB) and Random Partitioning(RP). It shows that our methods perform better than the other two algorithms as our algorithm uses effective data balancing technique and recursive based approach.

In Table 12 we present the comparison results for Min Average Precision. The base classifier is j48. We have compared the results with both of our techniques: Partition using Balanced Distribution(PDB) and Random Partitioning(RP). It is observed that our algorithms performs better than the other two algorithms.

In Table 13 we present the comparison results for Min Average Precision on the ten datasets. The base classifier is Naive Bayes. Here also, we have compared the results with both of our techniques: Partition using Balanced

TABLE 13. Min average precision comparison(base classifier: Naive Bayes).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	59.98	**60.10**	59.55	56.09
Dermatology	95.94	**97.29**	94.29	94.36
New thyroid	98.46	98.57	**99.70**	95.91
Wine	97.43	**97.70**	97.04	97.30
Splice	95.11	**95.34**	91.75	90.85
Led7digit	**79.88**	79.40	75.29	67.85
Zoo	**91.52**	88.77	87.91	90.29
Hayes Roth	**96.38**	95.76	90.81	90.88
Car	89.95	**89.99**	87.21	88.19
Satimage	**65.78**	65.65	64.75	63.89

TABLE 14. F measure comparison(base classifier: J48).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	**78.60**	77.50	58.44	71.77
Dermatology	97.14	**97.36**	96.93	95.21
New thyroid	**95.69**	95.45	93.10	93.06
Wine	97.93	**98.02**	93.68	95.70
Splice	95.22	**95.24**	92.64	92.18
Led7digit	**72.52**	72.33	71.22	66.86
Zoo	90.18	**90.30**	87.15	89.70
Hayes Roth	**86.00**	85.95	84.50	84.59
Car	97.65	**98.48**	97.10	83.74
Satimage	**77.98**	77.72	75.72	75.35

TABLE 15. F measure comparison(base classifier: Naive Bayes).

Dataset	PBD	RP	S.Bag -ging	SplitBal
Balance	75.89	**78.56**	75.55	72.23
Dermatology	96.12	**97.48**	93.92	94.64
New thyroid	**95.66**	95.61	94.50	92.91
Wine	97.97	**97.98**	97.90	95.90
Splice	95.16	**95.32**	93.06	92.23
Led7digit	72.49	**72.78**	69.31	67.01
Zoo	**91.34**	88.72	89.23	89.83
Hayes Roth	**76.37**	74.82	72.94	73.27
Car	**89.40**	89.06	87.53	83.90
Satimage	**77.50**	76.96	75.48	75.37

Distribution(PDB) and Random Partitioning(RP). It shows that our methods perform better than the other two algorithms as our methods use effective data balancing technique and recursive based approach.

In Table 14 we present the comparison results for F measure on the ten datasets. The base classifier is j48. We have compared the results with both of our techniques: Partition using Balanced Distribution(PDB) and Random Partitioning(RP). It shows that our methods perform better than the other two methods.

In Table 15 we present the comparison results for F measure on the ten datasets. The base classifier is Naive Bayes. We have compared the results with both of our techniques: Partition using Balanced Distribution(PDB) and Random Partitioning(RP). Here also, it shows that our algorithms performs better than the other two algorithms as our algorithms use effective data balancing technique and recursive based approach.

TABLE 16. t test results for average recall of RP with respect to SMOTEBagging and SplitBal(AVA)(base classifier: j48).

Dataset	With SMOTEBagging			With SplitBal(AVA)		
	DF	t value	Sig. Level	DF	t value	Sig. Level
Balance	9	0.891	0.2	9	2.797	0.025
Dermatology	9	1.437	0.1	9	9.487	0.0005
New Thyroid	9	1.139	0.15	9	2.109	0.05
Wine	9	2.121	0.05	9	2.637	0.05
Splice	9	4.596	0.001	9	11.805	0.0005
Zoo	9	1.304	0.15	9	1.382	0.15
Led7digit	9	3.311	0.005	9	3.339	0.005
Hayes Roth	9	1.440	0.1	9	0.770	0.25
Car	9	1.960	0.05	9	2.170	0.05
Satimage	9	1.15	0.15	9	6.880	0.0005

TABLE 17. t test results for average recall of RP with respect to SMOTEBagging and SplitBal(AVA)(base classifier: Naive Bayes).

Dataset	With SMOTEBagging			With SplitBal(AVA)		
	DF	t value	Sig. Level	DF	t value	Sig. Level
Balance	9	1.486	0.1	9	0.769	0.25
Dermatology	9	5.068	0.0005	9	2.86	0.01
New Thyroid	9	1.961	0.1	9	2.169	0.05
Wine	9	1.16	0.15	9	6.885	0.0005
Splice	9	8.783	0.0005	9	21.678	0.0005
Zoo	9	1.01	0.2	9	1	0.2
Led7digit	9	3.151	0.01	9	0.846	0.25
Hayes Roth	9	1.438	0.1	9	2.796	0.025
Car	9	1.138	0.15	9	9.488	0.0005
Satimage	9	3.312	0.005	9	3.340	0.005

7) t TEST FOR CLASSIFICATION

t-test is an well−known statistical technique which helps to discover if two models are notably different. Let $Model_1$ and $Model_2$ be two models having error rates $e(Model_1)_x$ and $e(Model_2)_x$ on round x. Mean error rate for $Model_1$, $\overline{er}(Model_1)$ is obtained by averaging the error rates of all the rounds. Likewise, $\overline{er}(Model_2)$ is also found. For N samples the t-statistic with $N-1$ degrees of freedom is computed. Here,

$$t = \frac{\overline{er}(Model_1) - \overline{er}(Model_2)}{\sqrt{var(Model_1 - Model_2)/N}}$$

where, $var(Model_1 - Model_2)$ indicates the variance of the difference between the two models.

To find out if $Model_1$ and $Model_2$ are notably different, t is computed. The significance level is then found from the t-table for the t-distribution.

Table 16 shows the t test results of our algorithm for multi-class imbalance problem using Random Partitioning with SMOTEBagging and SplitBal. The test is performed for Average Recall. The base classifier is j48. In all the datasets, the degree of freedom (DF) is 9. The results show that our algorithm is remarkably different from the other algorithms.

Table 17 also shows the t test results of our algorithm for multi-class imbalance problem using Random Partitioning with SMOTEBagging and SplitBal Algorithm. The test is performed for Average Recall. Here, the base classifier is Naive Bayes. In all the datasets, the degree of freedom (DF) is 9. The results show that our algorithm is significantly different from the other algorithms.

TABLE 18. Mean square error for regression analysis.

Dataset	Linear Regression	REPTree	LWR	Piecewise Regression	Balanced Regression
compactiv	0.30118	0.03388	0.07098	0.1368	**0.02042**
concrete	0.11539	0.08307	0.16525	0.09991	**0.06705**
delta-ail	0.13082	0.14135	0.16409	0.12699	**0.08484**
delta-elv	0.12855	0.13193	0.16865	0.12704	**0.09753**
diabetes	0.19487	0.21942	0.19293	0.18675	**0.18019**
ele-1	0.06014	0.06714	0.08328	0.06824	**0.05496**
elevators	0.09513	0.09264	0.19109	0.0771	**0.06897**
friedman	0.10596	0.10597	0.21149	**0.06444**	0.08095
laser	0.09539	0.05235	0.12432	0.0698	**0.041**
plastic	0.4283	0.17345	0.31458	0.18267	**0.14362**
tic	0.64137	0.6429	0.65574	0.63984	**0.53042**
autoMPG6	0.0794	0.06926	0.07011	0.07001	**0.06704**
ele-2	0.02317	0.01086	0.01003	0.010116	**0.00998**
house	0.09586	0.07562	0.0765	0.075001	**0.07153**
quake	0.20561	0.20515	0.20134	0.20063	**0.18664**
wankara	0.01554	0.02457	0.015008	0.015436	**0.01394**

TABLE 19. Mean absolute error for regression analysis.

Dataset	Linear Regression	REPTree	LWR	Piecewise Regression	Balanced Regression
compactiv	0.21557	0.01328	0.0344	0.0943	**0.01043**
concrete	0.09239	0.06342	0.1308	0.07975	**0.04634**
delta-ail	0.09035	0.09486	0.11322	0.08734	**0.05661**
delta-elv	0.09472	0.09558	0.12043	0.09368	**0.06501**
diabetes	0.16342	0.18281	0.16108	0.15747	**0.14822**
ele-1	0.04105	0.04564	0.06458	0.04986	**0.03629**
elevators	0.0647	0.06763	0.14434	**0.04849**	0.04934
friedman	0.08642	0.08828	0.19836	**0.05223**	0.06089
laser	0.06157	0.02838	0.09381	0.02586	**0.01933**
plastic	0.40027	0.13906	0.28402	0.1316	**0.07763**
tic	0.47868	0.47589	0.48749	0.47613	**0.30613**
autoMPG6	0.10158	0.09644	0.10012	0.100011	**0.09244**
ele-2	0.02686	0.01632	0.01756	0.01611	**0.01467**
house	0.15814	0.13178	0.13009	0.13465	**0.11802**
quake	0.25055	0.25037	0.24856	0.24345	**0.21694**
wankara	0.02034	0.03222	0.020103	0.020112	**0.01848**

B. PERFORMANCE ANALYSIS FOR DATA IMBALANCE REGRESSION

For evaluating and comparing the performance of our algorithm for regression for imbalance data, we have performed experiments on 16 real life datasets. We have found the datasets from Keel Dataset Respiratory. The datasets are: compactiv, concrete, delta-ail, delta-elv, diabetes, ele-1, elevators, friedman, laser, plastic, tic,autoMPG6, ele-2, house, quake, wankara. The performance measures used for evaluating their performance are Mean Absolute Error and Mean Square Error. For all the experiments, the $recursion_{limit}$ is set to 2. A higher value of this threshold will yield a better accuracy, but at the cost of more time. Since increasing the threshold would trigger an exponential increase in the number of recursive calls, it may not be feasible to set it to a large value. As there is a trade-off between accuracy and execution time, we experimented with the threshold and observed that setting it to 2 enables a good balance between the accuracy and efficiency. The *Imbalance Threshold* is set to 1.5 for our algorithm.

1) MEAN SQUARE ERROR

In this section we have presented the result of Mean Square Error. As our base regression model, we have used REPTree. We have compared our result with four regression algorithms (Linear Regression, REPTree, Locally Weighted Regression and Piecewise Regression). In all the cases, the error value is normalized so that it can have value ranging from 0-1. Table 18 shows the results of our algorithm and also the other three algorithms. It shows that our method performed better that the other four methods as we handle the data imbalance problem with an effective recursive based approach.

2) MEAN ABSOLUTE ERROR

In this section we have presented the result of Mean Absolute Error. As our base regression model we have used REPTree. Here also, we have compared our result with four regression algorithms (Linear Regression, REPTree, Locally Weighted Regression and Piecewise Regression). In all the cases, the error value is normalized so that it can have value ranging from 0-1. Table 19 shows the results of our algorithm and also the other two algorithms. It is observed that our method performed better than the other four methods as our algorithm handles the data imbalance problem with an effective recursive based approach.

3) t TEST FOR REGRESSION

For further establishing the efficacy of our algorithm, we have performed the t test for proving that our method is remarkably different from the other methods.

TABLE 20. t test results of Our balanced regression with respect to REPTree.

Dataset	DF	t value	Sig. Level
compactiv	9	8.434	0.0005
concrete	9	6.374	0.0005
delta-ail	9	4.85	0.0005
delta-elv	9	8.78	0.0005
diabetes	9	2.44	0.025
ele-1	9	1.856	0.05
elevators	9	12.419	0.0005
friedman	9	2.708	0.025
laser	9	4.297	0.001
plastic	9	4.781	0.0005
tic	9	4.778	0.0005
autoMPG6	9	7.553	0.0005
ele-2	9	6.765	0.0005
house	9	5.324	0.0005
quake	9	6.998	0.0005
wankara	9	1.857	0.05

TABLE 21. t test results of our balanced regression with respect to piecewise regression.

Dataset	DF	t value	Sig. Level
compactiv	9	9.569	0.0005
concrete	9	4.038	0.005
delta-ail	9	3.512	0.005
delta-elv	9	3.261	0.005
diabetes	9	1.772	0.01
ele-1	9	2.397	0.025
elevators	9	0.883	0.2
friedman	9	0.769	0.75
laser	9	4.158	0.005
plastic	9	5.221	0.0005
tic	9	12.314	0.0005
autoMPG6	9	0.770	0.75
ele-2	9	4.785	0.0005
house	9	5.334	0.0005
quake	9	1.875	0.05
wankara	9	6.665	0.0005

Table 20 shows the t test results of our algorithm for data imbalance regression with respect to REPTree. The test is performed for Mean Square Error. The degree of freedom is 9. From the results it is observed that our algorithm is notably different from the other algorithm.

Table 21 shows the t test results of our algorithm for data imbalance regression with respect to Piecewise Regression. The test is performed for Mean Square Error. Here also, the degree of freedom (DF) is 9. From the results it is clear that our algorithm is remarkably different from the other algorithm.

V. CONCLUSION

The nature of real life data is to be imbalanced. Many algorithms exist for data classification but most of them are hindered by the imbalance problem. Moreover, nearly all algorithms for class imbalance classification are mostly designed for binary classification. Algorithms like sampling change the data distribution of the original class which can cause problems during training. At every iteration of Bagging and Boosting algorithms, the sampled data has almost same class distribution with the original data, which is a potential cause of errors. All of these worked as our motivation for proposing efficient algorithms for multi-class imbalance classification and regression for imbalance data. In summary, we have introduced an effective recursive technique to correctly classify imbalanced data and extended that method to regression analysis.

For future work, we plan to introduce a more effective ensemble rule suitable for multi-class classification and regression, thereby improving our method. The proposed method is external, but we also plan to experiment and combine internal methods with it so it can be more effective in solving the imbalance problem.

REFERENCES

[1] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 2000.

[2] T. Alam, C. F. Ahmed, S. A. Zahin, M. A. H. Khan, and M. T. Islam, "An effective ensemble method for multi-class classification and regression for imbalanced data," in *Proc. Ind. Conf. Data Mining*, New York, NY, USA, Jul. 2018, pp. 59–74.

[3] S. Kulluk, L. Ozbakir, and A. Baykasoglu, "Training neural networks with harmony search algorithms for classification problems," *Eng. Appl. Artif. Intell.*, vol. 25, no. 1, pp. 11–19, Feb. 2012.

[4] R. Saraçoğlu, "Hidden Markov model-based classification of heart valve disease with PCA for dimension reduction," *Eng. Appl. Artif. Intell.*, vol. 25, no. 7, pp. 1523–1528, Oct. 2012.

[5] P. Donmez, *Introduction to Machine Learning*, 2nd ed. Cambridge, MA, USA: MIT Press 2010.

[6] Q. Yang and X. Wu, "10 challenging problems in data mining research," *Int. J. Inf. Technol. Decis. Making*, vol. 5, no. 4, pp. 597–604, Dec. 2006.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, pp. 321–357, Jun. 2002.

[8] S. Wang and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," in *Proc. IEEE Symp. Comput. Intell. Data Mining*, Mar./Apr. 2009, pp. 324–331.

[9] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *Proc. Int. Conf. Artif. Intell.*, vol. 1, 2000, pp. 111–117.

[10] J. Zhang and I. Mani, "KNN approach to unbalanced data distributions: A case study involving information extraction," in *Proc. Workshop Learn. Imbalanced Datasets II (ICML)*, Washington, DC, USA, 2003. [Online]. Available: https://www.site.uottawa.ca/~nat/Workshop2003/workshop2003.html

[11] Z. Sun, Q. Song, X. Zhu, H. Sun, B. Xu, and Y. Zhou, "A novel ensemble method for classifying imbalanced data," *Pattern Recognit.*, vol. 48, no. 5, pp. 1623–1637, May 2017.

[12] A. Estabrooks, T. Jo, and N. Japkowicz, "A multiple resampling method for learning from imbalanced data sets," *Comput. Intell.*, vol. 20, no. 1, pp. 18–36, Feb. 2004.

[13] A. Estabrooks and N. Japkowicz, "A mixture-of-experts framework for learning from imbalanced data sets," in *Advances in Intelligent Data Analysis* (Lecture Notes in Computer Science), vol. 2189. Berlin, Germany: Springer, 2001, pp. 34–43.

[14] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.

[15] W. Liu and S. Chawla, "Class confidence weighted *k*NN algorithms for imbalanced data sets," in *Advances in Knowledge Discovery and Data Mining* (Lecture Notes in Computer Science), vol. 6635, J. Z. Huang, L. Cao, and J. Srivastava, Eds. Berlin, Germany: Springer, 2011, pp. 345–356.

[16] J.-F. Díez-Pastor, J. J. Rodríguez, C. I. García-Osorio, and L. I. Kuncheva, "Random balance: Ensembles of variable priors classifiers for imbalanced data," *Knowl.-Based Syst.*, vol. 85, pp. 96–111, Sep. 2015.

[17] Y. Li, H. Guo, X. Liu, Y. Li, and J. Li, "Adapted ensemble classification algorithm based on multiple classifier system and feature selection for classifying multi-class imbalanced data," *Knowl.-Based Syst.*, vol. 94, pp. 84–104, Feb. 2016.

[18] Z. Zhang, B. Krawczyk, S. Garcìa, A. Rosales-Pérez, and F. Herrera, "Empowering one-vs-one decomposition with ensemble learning for multi-class imbalanced data," *Knowl.-Based Syst.*, vol. 106 pp. 251–263, Aug. 2016.

[19] H. Guo, Y. Li, Y. Li, X. Liu, and J. Li, "BPSO-Adaboost-KNN ensemble learning algorithm for multi-class imbalanced data classification," *Eng. Appl. Artif. Intell.*, vol. 49, pp. 176–193, Mar. 2016.
[20] W. Feng, W. Huang, and W. Bao, "Imbalanced hyperspectral image classification with an adaptive ensemble method based on SMOTE and rotation forest with differentiated sampling rates," *IEEE Geosci. Remote Sens. Lett.*, to be published.
[21] W. Feng, W. Huang, and J. Ren, "Class imbalance ensemble learning based on the margin theory," *Appl. Sci.*, vol. 8, no. 5, p. 815, 2018.
[22] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: One-sided selection," in *Proc. 14th Int. Conf. Mach. Learn.*, San Mateo, CA, USA: Morgan Kaufmann, 1997, pp. 179–186.
[23] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, Aug. 1996.
[24] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 42, no. 4, pp. 463–484, Jul. 2012.
[25] V. López, A. Fernández, M. J. del Jesus, and F. Herrera, "A hierarchical genetic fuzzy system based on genetic programming for addressing classification with highly imbalanced and borderline data-sets," *Knowl.-Based Syst*, vol. 38, pp. 85–104, Jan. 2013.
[26] Y. Freund, "Boosting a weak learning algorithm by majority," *Inf. Comput.*, vol. 121, no. 2, pp. 256–285, Sep. 1995.
[27] P. M. Domingos, "MetaCost: A general method for making classifiers cost-sensitive," in *Proc. 5th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, San Diego, CA, USA, Aug. 1999, pp. 155–164.
[28] C. Seiffert, T. M. Khoshgoftaar, and J. Van Hulse, "Improving software-quality predictions with data sampling and boosting," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 39, no. 6, pp. 1283–1294, Nov. 2009.
[29] N. Lavrac, D. Gamberger, H. Blockeel, and L. Todorovski, Eds., *Knowledge Discovery in Databases: PKDD 2003, 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia* (Lecture Notes in Computer Science), vol. 2838. Berlin, Germany: Springer, 2003.
[30] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 40, no. 1, pp. 185–197, Jan. 2010.
[31] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 39, no. 2, pp. 539–550, Apr. 2009.
[32] M. Galar, A. Fernández, E. Barrenechea, and F. Herrera, "EUSBoost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling," *Pattern Recognit.*, vol. 46, no. 12, pp. 3460–3471, Dec. 2013.
[33] B. Krawczyk and G. Schaefer, "An improved ensemble approach for imbalanced classification problems," in *Proc. IEEE 8th Int. Symp. Appl. Comput. Intell. Inform. (SACI)*, Timisoara, Romania, May 2013, pp. 423–426.
[34] B. Krawczyk, M. Woźniak, and F. Herrera, "Weighted one-class classification for different types of minority class examples in imbalanced data," in *Proc. IEEE Symp. Comput. Intell. Data Mining (CIDM)*, Orlando, FL, USA, Dec. 2014, pp. 337–344.
[35] R. C. Prati, G. E. A. P. A. Batista, and D. F. Silva, "Class imbalance revisited: A new experimental setup to assess the performance of treatment methods," *Knowl. Inf. Syst.*, vol. 45, no. 1, pp. 247–270, Oct. 2015.
[36] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Hong Kong, Jun. 2008, pp. 1322–1328.
[37] A. Fernández, V. López, M. Galar, M. J. del Jesus, and F. Herrera, "Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches," *Knowl.-Based Syst.*, vol. 42, pp. 97–110, Apr. 2013.
[38] K. Buza, A. Nanopoulos, and G. Nagy, "Nearest neighbor regression in the presence of bad hubs," *Knowl.-Based Syst.*, vol. 86, pp. 250–260, Sep. 2015.
[39] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "Classification and regression trees," Wadsworth Int., Monterey, CA, USA, Tech. Rep., 1984. [Online]. Available: https://content.taylorfrancis.com/books/download?dac=C2009-0-07054-X&isbn=9781351460491&format=googlePreviewPdf
[40] L. Torgo, P. Branco, R. P. Ribeiro, B. Pfahringer, "Resampling strategies for regression," *Expert Syst.*, vol. 32, no. 3, pp. 465–476, Jun. 2015.
[41] P. Branco, L. Torgo, and R. Ribeiro, "A survey of predictive modelling under imbalanced distributions," May 2015, *arXiv:1505.01658*. [Online]. Available: https://arxiv.org/abs/1505.01658
[42] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning," in *Proc. Int. Conf. Intell. Comput.*, Hefei, China, Aug. 2005, pp. 878–887.
[43] J. Xie, and Z. Qiu, "The effect of imbalanced data sets on LDA: A theoretical and empirical analysis," *Pattern Recognit.*, vol. 40, no. 2, pp. 557–562, Feb. 2007.

TAHIRA ALAM received the B.S. and M.S. degrees in computer science and engineering from the Department of Computer Science and Engineering, University of Dhaka, Bangladesh, in 2015 and 2017, respectively, where she is currently involved in research with the Data Mining Research Group, Department of Computer Science and Engineering. She is also a Lecturer with the University of Asia Pacific, Bangladesh. Her research interests include machine learning and data mining.

CHOWDHURY FARHAN AHMED received the B.S. and M.S. degrees in computer science from the University of Dhaka, Bangladesh, in 2000 and 2002, respectively, and the Ph.D. degree in computer engineering from Kyung Hee University, South Korea, in 2010. He was a Postdoctoral Research Fellow with the ICube Laboratory, University of Strasbourg, France, from 2013 to 2015. Since 2004, he has been a Faculty Member with the Department of Computer Science and Engineering, University of Dhaka, Bangladesh, where he is currently a Professor. His research interests include databases, data mining, knowledge discovery, and machine learning.

SABIT ANWAR ZAHIN received the B.S. and M.S. degrees in computer science and engineering from the Department of Computer Science and Engineering, University of Dhaka, Bangladesh, in 2015 and 2017, respectively. He is currently an Active Member of the Data Mining Research Group, Department of Computer Science and Engineering, University of Dhaka. He is also a Research Intern with NewsCred. His research interests include artificial intelligence, data mining, machine learning, and games.

MUHAMMAD ASIF HOSSAIN KHAN received the B.Sc. and M.S. degrees in computer science from the University of Dhaka, Bangladesh, in 1999 and 2001, respectively, and the Ph.D. degree from The University of Tokyo, Japan, in 2014. He joined the University of Dhaka as a Faculty Member in 2002. He is currently a Professor with the Department of Computer Science and Engineering, University of Dhaka, where he is also the Director of the ICT Cell. His research interests include natural language processing, information retrieval, microblog analysis, and machine learning.

MALIHA TASHFIA ISLAM received the B.S. degree in computer science and engineering from the Department of Computer Science and Engineering, University of Dhaka, Bangladesh, in 2018, where she is currently involved in research with the Data Mining Research Group. Her research interests include data mining and cyber security.