Biased Random Forest For Dealing With the Class Imbalance Problem

Mohammed Bader-El-Den, Eleman Teitei, and Todd Perry

***Abstract*— The class imbalance issue has been a persistent problem in machine learning that hinders the accurate predictive analysis of data in many real-world applications. The class imbalance problem exists when the number of instances present in a class (or classes) is significantly fewer than the number of instances belonging to another class (or classes). Sufficiently recognizing the minority class during classification is a problem as most algorithms employed to learn from data input are biased toward the majority class. The underlying issue is made more complex with the presence of data difficult factors embedded in such data input. This paper presents a novel and effective ensemble-based method for dealing with the class imbalance problem. This paper is motivated by the idea of moving the oversampling from the data level to the algorithm level, instead of increasing the minority instances in the data sets, the algorithms in this paper aims to "oversample the classification ensemble" by increasing the number of classifiers that represent the minority class in the ensemble, i.e., random forest. The proposed biased random forest algorithm employs the nearest neighbor algorithm to identify the critical areas in a given data set. The standard *random forest* is then fed with more random trees generated based on the *critical areas*. The results show that the proposed algorithm is very effective in dealing with the class imbalance problem.**

***Index Terms*— Class imbalance, classification, nearest neighbor, random forest.**

I. Introduction

CLASSIFICATION algorithms have aided the analysis and prediction of data in many real-world application domains. However, learning algorithms encounter difficulties of assigning correct labels to instances when learning from imbalanced data distribution schemes, this is generally referred to as the "class imbalance problem." The class imbalance problem exists when a class(es) commonly referred to as the minority class(es) is under represented when compared against the other class(es), also known as the majority class(es). Such scenario exists in many real-life applications [1], [2].

Several approaches initiated to deal with the imbalance problem ranging from externally rebalancing the class distribution to internally tweaking the learning algorithms to adapt to the imbalanced nature, have all accentuated success in varying degrees. Notwithstanding, some researchers have argued that learning algorithms adequately learn from the minority class when they are linearly separable from their majority counterparts [3]. The points at which the minority instances are positioned within the majority instances in an imbalance scheme contributes to the increase in misclassification rate, thus commonly referred to as data difficult factors [4], but will be referred to as *critical* or *difficult areas* in this paper. These factors include, but are not limited to: small disjuncts, class overlap, borderline, noise, outliers, and rare instances [1].

This paper is motivated by the idea of moving the oversampling from the data level to the algorithm level. In other words, instead of increasing the minority instances, the proposed algorithms aims to "oversample the ensemble" by increasing the number of classifiers that represent the minority class in the ensemble (random forest in this paper). Therefore, this paper introduces a hybrid ensemble method—biased random forest (BRAF)—aimed at adequately representing the minority class during classification. BRAF employs the nearest neighbor algorithm to identify the difficult areas in the data set which are the minority instances with their k-nearest majority neighbors. The standard *random forest* is then fed with more random trees generated based on the *critical* or *difficult areas*, resulting in a more diverse ensemble/forest and at the same time biased toward the minority class. The bias in the forest aims to overcome the low presence of the minority class(es).

> *"Given a data set with imbalanced classes, can extending the decision trees of a random forest to not only learn from the original training set but also from the* critical/difficult areas *of a given imbalanced data set improve the recognition of the minority instances during classification? How could these* difficult areas *be discovered? What is the impact of feeding these trees on the ensemble diversity?* Thus, the objectives of this paper are as follows.

1) Design a hybrid framework that generates an ensemble that is biased toward the minority class(es).
2) Select a mechanism for defining *difficult/critical areas* in the input data set.
3) Evaluate and compare BRAF against other state of the art methods using several artificial and real-world binary class-imbalanced data sets.
4) Evaluate the diversity of the new biased forests.

The rest of this paper is structured as follows. Section II briefly explains and reviews some of the related state-of-the-art

Manuscript received November 6, 2017; revised May 28, 2018 and October 14, 2018; accepted October 15, 2018. Date of publication November 20, 2018; date of current version June 14, 2019. *(Corresponding author: Mohamed Bader-El-Den.)*

M. Bader-El-Den and E. Teitei are with the School of Computing, University of Portsmouth, Portsmouth PO1 3HE, U.K. (e-mail: mobahy@gmail.com; eleman.teitei@port.ac.uk).

T. Perry is with the Research and Development, Huq Industries, London WC1N 3QA, U.K.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2018.2878400

approaches developed to tackle the class imbalance problem. Section III describes the proposed approach, BRAF. Section IV encompasses various measures used in this paper for evaluating BRAF's performance and its experimental outcomes. Finally, Section V summarizes this paper.

II. Background

A. Dealing With Imbalance Problems

Methods employed to tackle class imbalance are generally categorized as follows: data-level approaches that aim to reduce the imbalanced data ratio by adding more minority instances (oversampling) or discarding some of the majority instances (undersampling) [5]. These techniques are mostly employed during the data preprocessing phases and are independent of the succeeding learning algorithms. Approaches at the algorithmic level aim at internally modifying algorithms used for classification to sufficiently learn from the minority instances [6]. A detailed understanding of the corresponding algorithm and domain is required when employing methods from this category. Algorithmic approaches include, but are not limited to one class learning [7], and changing the internal bias [8], [9]. Also, classified as a category is cost-sensitive learning which could be referred to as a hybrid approach, as it integrates the inclusion of cost to instances (data level), and adjusting the succeeding training processes to accept cost (algorithmic level) [10]. Furthermore, ensemble learning methods learn from data by employing several base classifiers and systemically combining their outcome to produce a single predictive decision.

While the existing methods have emanated varying degrees of success when applied in imbalanced data schemes, this paper concentrates more on the data level and the ensemble categories. The reason is because the motivation of BRAF's concept was deduced from both categories.

Most data-level approaches embed the nearest neighbor rule (NNR) concept which emanates from the hypotheses on the role of the mutual positions of the learning instances in the feature space. Some of such data-level approaches include but are not limited to: synthetic minority-sampling technique (SMOTE). It creates synthetic minority instances from a chosen point of the line that links a selected minority instance (seed) to its nearest minority neighbor [11]. Borderline-SMOTE creates artificial instances from only those minority instances that are located near the decision border [12]. Safe-level SMOTE [13] and local neighborhood SMOTE [14] generate synthetic minority instances only from examples located in safe areas. Furthermore, adaptive synthetic sampling generates more synthetic instances from the identified difficult regions and lesser instances from safe areas [15]. Also, edited nearest neighbor (ENN) discards majority instances where their two or three nearest neighbors belong to the minority class [16]. Nearest cleaning rule is similar to ENN, but further discards majority instances that surround a minority instance [17].

B. Random Forest

Ensemble learning systemically combines the learning outcomes of several base classifiers so as to deduce a single prediction that outperforms the outcome of individual base classifiers [1]. The underlying concept centers around analyzing the opinion of different individuals so as to deduce a single sound opinion. An ensemble learning method's effectiveness is mainly dependent on the distinctiveness of the individual base classifiers. Two popular ensemble learning techniques are *Bagging* (Bootstrap Aggregation) [18] and *Boosting* [19]. In bagging, classifiers are trained in parallel on each bootstrap sample generated from the training set. The predictive outcomes made by each classifier are systematically analyzed and a single decision is deduced using majority voting. On the other hand, boosting trains classifiers in a sequential manner. Each succeeding classifier is trained on a reweighted data subset that was misclassified in the preceding phase. A popular boosting method is AdaBoost [20].

Random forest is a learning algorithm developed by Breiman [21], which conforms to the bagging concept. It is an ensemble of decision trees; each tree representing a base classifier. Its classification process is implemented by taking a majority vote of the predictions that emanates from the individual trees trained on data subsets generated from the parent training set [22].

C. Related Work

Several methods that entail the modification of the random forest learning concept have been proposed to deal with the class imbalance problem. Chen [23] proposed weighed random forest (WRF) which embeds the cost-sensitive learning concept. It assigns weight to both classes (minority and majority) of a training set; the minority instances having a larger weight. The weights are embedded in two phases. First to calculate the gini criterion when finding the split of instances. Second, class weights are fixed in each decision tree's terminal node which aids in determining the weighted majority vote. The weighted vote of each decision tree is aggregated so as to deduce a conclusive forest prediction model.

Chen [23] also introduced a balanced random forest (BRF) which incorporates a sampling technique. BRF was proposed to tackle the possibility that some of the generated bootstrap samples might contain fewer or none of the minority instances. The underlying idea of BRF is to systematically undersample the majority class during the generation of bootstrap samples. Furthermore, Chin *et al.* [23] proposed imbalanced BRF (IBRF) that systematically combined both BRF and WRF while trying to tackle the churn prediction issue in Chinese banks. IBRF's underlying idea was aimed at harnessing the strengths of WRF and BRF. While BRF is more tolerant to noise and also more efficient on large imbalanced data schemes, the cost-sensitive learning embeded in WRF proved to be more effective on classifiers that emanate from decision tree learning methods.

Another related concept employed to tackle class imbalance is UnderBagging [24]. UnderBagging is a combination of undersampling and bagging. The underlying idea can be implemented in two ways. Either the majority class is undersampled before bagging is applied or it is undersampled in each of

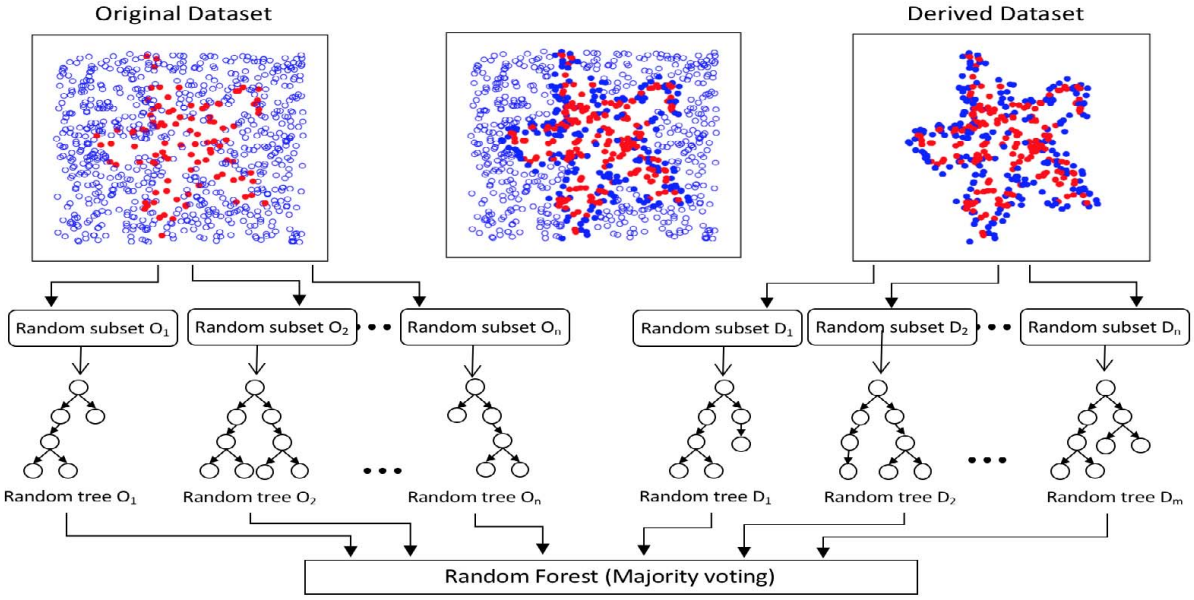

Fig. 1. Pictorially illustrates BRAF's concept for a binary class imbalance scheme. The *difficult/critical areas* of the Subclus70 data set are identified using $k = 5$ when defining the nearest majority neighbor parameter. Bootstrap samples are generated from the original training set, T as well as the derived training set, T_c. The random trees of the forest are then employed to not only learn from T but also extended to learn from the derived data input T_c.

the bootstrap sample generation stages. The UnderBagging concept has been applied to varying degrees using dissimilar titles such as QuasiBagging [25], asymmetric bagging [26], and roughly-balanced bagging [27].

The minority class, which is the point of interest, is always under represented in any imbalanced data scheme. A similar scenario is also apparent during the random forest learning process. There exists a high probability that fewer or no minority instances will be present in the generated bootstrap samples, which in-turn, contributes to the insufficient recognition of the minority class [23].

III. Biased Random Forest

This section presents the BRAF which is a simple and effective method that can be combined with any ensemble method. BRAF is motivated by the idea of moving the oversampling from the data level to the algorithm level. In other words, instead of increasing the minority instances in the data set, the algorithm in this paper aims to "oversample the classification ensemble" by increasing the number of classifiers that represent the minority class in the ensemble. This oversampling of the classification ensemble aims to generate an ensemble biased toward the minority class to compensate its low presence in the data set.

The overall structure of the BRAF algorithm is shown in Fig. 1. First, the nearest neighbor algorithm is employed by the BRAF algorithm to identify the *difficult/critical areas* in the data set, which are the minority instances and their k-nearest majority neighbors. Second, a standard random forest is generated from all the records in the data set. Third, the standard random forest is then fed with more random trees generated based on the *difficult areas*, hopefully resulting in a more diverse ensemble/forest and at the same time, biased toward the minority class. The bias in the forest aims to overcome the low presence of instances belonging to the minority class(es).

Another interesting way of looking at BRAF is to consider the first step in which the *difficult areas* are defined as an aggressive undersampling of the main data input using k-nearest neighbor algorithm. In this scenario, an undersampled subtraining set that contains all the minority instances and their k-nearest majority neighbors are generated from the original training set. Then, the random forest's decision trees are employed to learn from the bootstrap samples generated from the original training set, as well as those generated from the subtraining set.

The BRAF algorithm is detailed in Algorithm 1; given a binary imbalanced training set T, BRAF first divides the data set into: (a) a majority set T_{maj} containing the records from the majority class(es) and (b) a minority set T_{min} containing the records from the minority class(es). Then, the potential *difficult areas* affecting the minority instances are defined and stored in T_c. This is done by finding the k-nearest majority neighbors for each record in T_{min}. The k-nearest majority neighbors shared by two or more minority instances are extracted only once so as to avoid duplication. The decision trees of the forest are applied to not only learn from the bootstrap samples generated from T but are also extended to learn from those generated from T_c; this is done by combing two separate *random forests*. The first forest RF_1 is generated based on the full data set, T while the second forest RF_s is generated based on the under-sampled data set, T_c. BRAF uses two parameters to define the size of the generated random forest. The first parameter is s which defines the size of the combined forest, while the second parameter is p, where $0 \leq p \leq 1$ which is the ratio used to define the size of RF_2 and RF_1, i.e., the size of $RF_2 = p*s$ and consequently the size of $RF_1 = (1-p)*s$.

Algorithm 1 BRAF

```
Read the training data set T.
Majority set of labels is L_maj.
Minority set of labels is L_min.
Read forest total size S
Read p Ration { The ratio of the critical areas RF}
for each t_i in T do
  if label(t_i) ∈ L_maj then
    Add t_i to T_maj {Split T into minority T_maj and majority T_min sets.}
  else
    Add t_i to T_min
  end if
end for
for each t_i in T_min do
  add t_i to T_c {Adds the t_i to the critical data set.}
  T_nn = Nearest_Neighbor(t_i, T_maj, k) {Find the k nearest neighbor for each minority instance in the data set.}
  for each t_j in T_nn do
    if t_j ∉ T_c then
      add t_j to T_c {Add the unique neighbors only to the critical data set.}
    end if
  end for
end for
RF_1 = BuildForest(T, S × (1 − p)) {Build a first forest based on the full data set of size S × (1 − p).}
RF_1 = BuildForest(T_c, S × p) {Build a first forest based on the critical data set data set of size S × p.}
RF = RF1 + RF2 {Combine the two forests to generate the main forest RF.}
```

In the training phase, a random forest applies the bagging concept to iteratively generate subtraining sets (bootstrap samples) with replacements. The generated bootstrap samples contain the same number of records present in their parent training sets. These records are chosen at random. An average of 64% of instances in the parent training sets is available in the bootstrap samples. Randomly choosing instances accentuates a high possibility that some instances will be repeated multiple times while some might not appear at all. This is the point where BRAF's effectiveness is accentuated as the generated bootstrap samples are likely to contain fewer or none of the minority instances. BRAF's concept serves as a boost to enhance random forest's capability to effectively recognize the minority instances in the sense that extracting bootstrap samples from T_c (depending on the chosen k-parameter) increases the chances of more minority instances to be present, thereby improving the accuracy of the outcome of the individual trees. This is controlled by the parameter p; the high value of p increases the number of decision trees learning from the *difficult/critical areas* T_c or decreases the number of those learning from the full training set T. The prediction of each individual tree is systematically analyzed to produce a single outcome by applying majority voting. The built classification model is further used to assign labels to unlabelled instances belonging to the test set Ts.

IV. Experimental Study

This section shows the performance of the proposed BRAF method. The aim of this section is as follows.

1) Analyze the performance of BRAF using different parameters, i.e., k the number of neighbors and p the ratio of *difficult areas* T_c.
2) Show the performance gain of BRAF when compared to the standard assemble methods.
3) To compare the performance of BRAF with similar methods, in particular, data-level oversampling methods (BRAF could be seen as a method for over sampling the assignable instead oversampling the data).
4) Analyze the diversity of the proposed BRAF methods, as diversity has been used to explain the success of several ensemble methods, theoretically and practically [28].

A. Evaluative Performance Metrics

In classification, accuracy is commonly used as a performance metric for measuring the percentage of correct predictions made. Accuracy has, however, proven not to be a suitable metric to be exerted in a class imbalance scenario as it is biased toward the majority instances and maintains a high percentage of accuracy even when all the minority examples are misclassified. Being that the minority class is the point of interest, other performance measurement metrics such as receiver operating characteristic, area under curve, F-measure (FM), and G-mean which is adopted in this paper.

B. Data Sets

Experiments were carried out using 21 imbalanced binary data sets. 12 of these data sets are artificial data sets from the Knowledge Extraction based on Evolutionary Learning (KEEL) repository [29]. They all have 800 examples

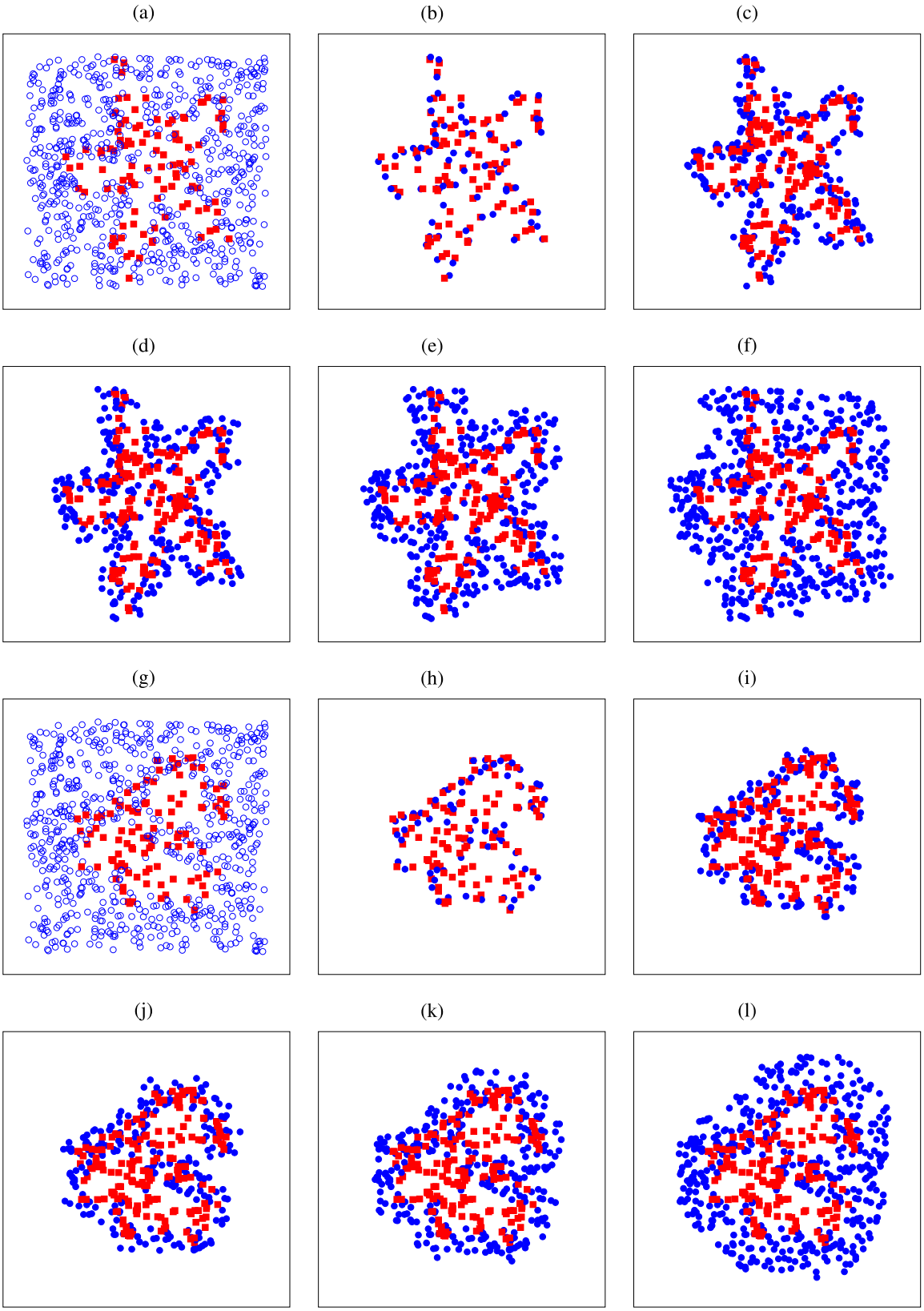

Fig. 2. Visualizes the impact of the number of majority neighbors ($k = \{1, 5, 10, 20, 50\}$) on the data set. The figures shows the Colver70 and Paw70 as these are artificial data sets with only two attributes which make it visualisable. The x and y axes are the value of the first and second attributes, respectively. (a) Clover70—original set. (b) $k = 1$. (c) $k = 5$. (d) $k = 10$. (e) $k = 20$. (f) $k = 50$. (g) Paw70—Original set. (h) $k = 1$. (i) $k = 5$. (j) $k = 10$. (k) $k = 20$. (l) $k = 50$.

with an imbalance ratio (IR) of 1:7. Their majority instances were invariably spread around the minority instances taking the shapes of a paw, several subclusters, and a clover as depicted in Fig. 2. The minority instances in the paw data

TABLE I
DESCRIPTION OF THE BINARY IMBALANCED DATA SETS USED IN THIS PAPER

	Dataset	#Instances	#Features	# Min	#Maj	#IR
Artificial	Clover0	800	2	100	700	7
	Clover30	800	2	100	700	7
	Clover50	800	2	100	700	7
	Clover70	800	2	100	700	7
	Paw0	800	2	100	700	7
	Paw30	800	2	100	700	7
	Paw50	800	2	100	700	7
	Paw70	800	2	100	700	7
	Subclus0	800	2	100	700	7
	Subclus30	800	2	100	700	7
	Subclus50	800	2	100	700	7
	Subclus70	800	2	100	700	7
Real-World	Abalone19	4174	8	32	4142	129.44
	Breast	286	9	85	201	2.36
	Bupa	345	8	145	200	1.38
	Car-good	1728	6	69	1659	24.04
	Haberman	306	19	81	225	2.78
	Hepatitis	155	9	32	123	5.35
	Pima	768	8	268	500	1.87
	Poker	2075	10	25	2250	82
	Yeast	1484	8	51	1433	28.1

set were decomposed into three elliptic smaller regions with two of its regions situated near each other. The minority instances in the clover data set are organized in an order that portrays a flower with elliptic petals which make them nonlinear and more difficult for an algorithm to learn from. While in the subcluster data set, minority instances are placed in rectangular shapes which are uniformly surrounded by the majority instances. To increase their difficulty, the disturbance ratio of their underlying borderline and small disjunt examples were increased by 30%, 50%, and 70%, respectively.

Furthermore, nine real-world application data sets were downloaded from the University of California Irvine (UCI) Machine Learning Repository repository [30]. These data sets have varying degrees of IRs, and contain few or no safe minority examples [31]. For example, the minority examples in the Herbaman data set are made up of 10 safe, 21 outlier, and 51 borderline instances. See Table I for their description, including the number of instances, number of features, and IR.

C. Experimental Setup and Analysis

In order to analyze and understand the performance of the proposed BRAF method, different sets of experiments were carried out, where the performance of the BRAF was compared against the performance of some other state of the art methods. Also, we tested BRAF (with and without SMOTE) using different k-parameters. Furthermore, in order to understand the behavior of the BRAF, diversity analysis was also carried out. In this analysis, we measured BRAF's level of diversity between the decision trees as compared against a base random forest algorithm.

In all experiments, the algorithms were run using 10-fold cross validation. The *knn* employed ranged between 1 and 50, while 100 decision trees were interchangeably used to learn from each of the training sets. Each experiment was run 10× using 21 data sets. Geometric mean (GM) and FM were used to evaluate the classification performance. Furthermore, BRAF's diversity was evaluated using the dis-agreement measure [32]. The base random forest classifier parameters were tuned by evaluating the RF parameters, the same parameters have been used for both the RF and BRAF.

1) Performance Analysis: Table II shows the performance comparison between BRAF's k values and the random forest base classifier. BRAF's results were generated using the size of the forest $s = 100$ and $k = 1$, 10, and 20. In all the experiments $p = 0.50$, which means that half of the forest is build based on the original training data T, while the second half is based of the *difficult areas* data set T_c. G-mean and FM are used as evaluative measurement metrics. The results depicted are the average of 10 runs ± standard deviation. The best overall results for each data set are marked with (*). The aim of this experiment is to ascertain the effectiveness of different Ks applied in BRAF when selecting the nearest majority neighbors for the *difficult areas* set. A total of 19 data sets were used; 7 real-world and 12 artificial data sets from the KEEL and UCI repositories, respectively.

BRAF performed better when compared against the random forest base classifier across all data sets when using FM as a performance metric. However, when using Gmean, random forest performed slightly better than BRAF on Breast, Clover0, Haberman, and Pima data sets.

As mentioned earlier, the target of the experiments is not to show that BRAF outperform other state-of-the-art methods, instead the target is to: 1) show that it is possible to improve the performance of an assignable on imbalanced data sets by making it biased toward the minority class and 2) compare the performance of BRAF with similar methods, in particular, data-level oversampling methods (BRAF could be seen as method for over sampling the assignable instead oversampling the data). In order to achieve the second point, the BRAF was compared with SMOTE as it is a widely used over sampling method. SMOTE, as any sampling method is not a stand-alone classifier, but it can be used in conjunction with any classifier. Table III compares the performance of BRAF (with and without SMOTE) against the base classifier RF with SMOTE. Several configurations of SMOTE were tested and best settings were used for both RF and BRAF. Unlike RF, and as expected, BRAF did not perform well when combined with SMOTE as the data became more balanced. Moreover, as given in Table III, the performance of BRAF decreases with the increase in the number of neighbors k. However, the performance of the BRAF without SMOTE is very competitive and has outperformed RF+SMOTE on several occasions, especially on the GM measure.

BRAF's best results were also compared against other state-of-the-art methods that have proven to improve classification performance in difficult class imbalance scenarios. They include, SMOTE [11], TempC [33], and AdaBoost [20]. The experiment was carried out on nine real-world application data sets using G-mean as an evaluative performance metric. BRAF's best results were selected from $K = 1 - 50$. As can be deduced in Table IV, BRAF performed better than the random forest base classifier, SMOTE, and AdaBoost on all data sets except on Breast where AdaBoost performed slightly better. TempC was slightly competitive against BRAF as it performed better on the Bupa, Car, and Yeast data sets.

TABLE II

COMPARES BRAF'S PERFORMANCE ON A RANGE OF DATA SETS USING $k = 1$, 10, AND 20 AGAINST THE STANDARD RANDOM FOREST ALGORITHM USING G.MEAN AND FM AS EVALUATIVE PERFORMANCE METRICS. THE RESULTS ARE THE AVERAGE OF 10 RUNS OF 10 CROSS VALIDATION, ± THE STANDARD DEVIATION. THE BEST OVERALL RESULTS FOR EACH DATA SET ARE MARKED BY (*)

	Base		**BRAF1**		**BRAF10**		**BRAF20**	
	fm	gm	fm	gm	fm	gm	fm	gm
Breast	0.39 ± 0.17	**0.60 ± 0.14***	**0.40 ± 0.18***	0.56 ± 0.13	0.38 ± 0.19	0.59 ± 0.15	0.38 ± 0.19	0.58 ± 0.14
Bupa	0.60 ± 0.12	0.69 ± 0.08	**0.63 ± 0.10**	0.67 ± 0.07	**0.63 ± 0.10**	**0.71 ± 0.05***	**0.62 ± 0.10**	**0.70 ± 0.06**
Car	0.42 ± 0.20	0.74 ± 0.29	**0.58 ± 0.16***	**0.83 ± 0.12**	**0.55 ± 0.15**	**0.84 ± 0.14**	**0.57 ± 0.12**	**0.84 ± 0.12***
Clover0	0.59 ± 0.11	**0.93 ± 0.08***	**0.83 ± 0.07***	0.88 ± 0.07	**0.81 ± 0.08**	0.91 ± 0.07	**0.80 ± 0.07**	0.91 ± 0.07
Clover30	0.60 ± 0.09	0.64 ± 0.80	0.59 ± 0.11	**0.75 ± 0.10**	0.60 ± 0.11	**0.81 ± 0.09**	**0.64 ± 0.13***	**0.83 ± 0.10***
Clover50	0.47 ± 0.12	0.75 ± 0.13	0.43 ± 0.10	0.64 ± 0.12	**0.48 ± 0.09***	0.74 ± 0.11	**0.48 ± 0.11**	**0.75 ± 0.09***
Clover70	0.30 ± 0.08	0.61 ± 0.13	**0.39 ± 0.11***	0.59 ± 0.11	0.28 ± 0.08	**0.62 ± 0.12***	0.29 ± 0.09	**0.62 ± 0.13**
Haberman	0.39 ± 0.15	**0.58 ± 0.15***	**0.42 ± 0.12***	0.53 ± 0.12	0.34 ± 0.18	0.51 ± 0.22	0.37 ± 0.14	0.56 ± 0.14
Hepatitis	0.42 ± 0.33	0.57 ± 0.41	**0.56 ± 0.32***	**0.63 ± 0.34**	**0.52 ± 0.31**	**0.66 ± 0.36***	0.37 ± 0.35	0.51 ± 0.44
Paw0	0.91 ± 0.08	0.94 ± 0.05	0.90 ± 0.1	0.94 ± 0.07	**0.92 ± 0.07***	**0.96 ± 0.04***	0.91 ± 0.07	**0.95 ± 0.04**
Paw30	0.65 ± 0.15	0.40 ± 0.07	**0.68 ± 0.15**	**0.80 ± 0.11**	**0.69 ± 0.13**	**0.85 ± 0.08***	**0.71 ± 0.15***	**0.84 ± 0.09**
Paw50	0.57 ± 0.09	0.78 ± 0.08	**0.59 ± 0.09**	0.75 ± 0.09	**0.59 ± 0.09**	**0.79 ± 0.08**	**0.63 ± 0.07***	**0.82 ± 0.07***
Paw70	0.44 ± 0.17	0.74 ± 0.15	**0.49 ± 0.12***	0.69 ± 0.10	**0.47 ± 0.15**	**0.75 ± 0.15**	**0.49 ± 0.14**	**0.77 ± 0.14***
Pima	0.21 ± 0.07	**0.76 ± 0.04***	**0.66 ± 0.08**	0.71 ± 0.06	**0.66 ± 0.07**	0.74 ± 0.04	**0.67 ± 0.07***	0.75 ± 0.04
Poker	0.00 ± 0.00	0.00 ± 0.00	**0.47 ± 0.43***	**0.60 ± 0.51***	**0.21 ± 0.36**	**0.30 ± 0.48**	**0.04 ± 0.13**	**0.10 ± 0.31**
Subc0	0.96 ± 0.06	0.98 ± 0.03	0.96 ± 0.06	0.98 ± 0.03	0.96 ± 0.06	0.98 ± 0.03	0.96 ± 0.06	0.98 ± 0.03
Sub30	0.69 ± 0.07	0.87 ± 0.06	0.68 ± 0.1	0.83 ± 0.10	**0.7 ± 0.07***	**0.88 ± 0.05***	0.69 ± 0.08	**0.88 ± 0.07**
Subc50	0.41 ± 0.14	0.68 ± 0.13	**0.41 ± 0.11***	0.68 ± 0.12	0.40 ± 0.13	**0.69 ± 0.11***	**0.41 ± 0.12***	0.68 ± 0.12
Subc70	0.30 ± 0.12	0.59 ± 0.14	**0.44 ± 0.12**	**0.61 ± 0.09**	**0.40 ± 0.14**	**0.61 ± 0.13**	**0.40 ± 0.11***	**0.62 ± 0.09***

TABLE III

COMPARES BRAF'S PERFORMANCE USING DIFFERENT $k's$–$k = 1$, 10, AND 20—AGAINST THE SMOTE + STANDARD RANDOM FOREST ALGORITHM AND SMOTE + BRAF USING G.MEAN AND FM AS EVALUATIVE PERFORMANCE METRICS. THE RESULTS ARE THE AVERAGE OF 10 RUNS ± THE STANDARD DEVIATION. THE BEST OVERALL RESULTS FOR EACH DATA SET ARE MARKED BY (*)

	Base + SMOTE		BRAF1 + SMOTE		BRAF10 + SMOTE		BRAF10	
	FM	GM	FM	GM	FM	GM	FM	GM
Breast	0.43 ± 0.14	0.56 ± 0.11	**0.45 ± 0.13**	0.56 ± 0.13	**0.49 ± 0.14***	**0.59 ± 0.15**	0.38 ± 0.19	**0.59 ± 0.15**
Bupa	0.63 ± 0.09	0.67 ± 0.06	0.58 ± 0.08	0.02 ± 0.12	0.50 ± 0.10	0.45 ± 0.21	**0.63 ± 0.1***	**0.71 ± 0.05***
Car	**0.66 ± 0.11***	0.83 ± 0.08	0.58 ± 0.16	0.83 ± 0.12	0.55 ± 0.15	0.84 ± 0.14	0.55 ± 0.15	**0.84 ± 0.14***
Clover0	**0.84 ± 0.09**	**0.93 ± 0.07**	0.25 ± 0.07	0.30 ± 0.17	0.20 ± 0.11	0.30 ± 0.15	0.81 ± 0.08	0.91 ± 0.07
Clover30	**0.69 ± 0.09***	0.79 ± 0.08	0.04 ± 0.03	0.02 ± 0.05	0.00 ± 0.00	0.00 ± 0.00	0.60 ± 0.11	**0.81 ± 0.09***
Clover50	**0.56 ± 0.11***	0.72 ± 0.12	0.25 ± 0.07	0.33 ± 0.14	0.00 ± 0.03	0.01 ± 0.04	0.48 ± 0.09	**0.74 ± 0.11***
Clover70	**0.42 ± 0.09***	**0.64 ± 0.1***	0.24 ± 0.07	0.22 ± 0.20	0.00 ± 0.00	0.00 ± 0.00	0.28 ± 0.08	0.62 ± 0.12
Haberman	0.40 ± 0.17	**0.54 ± 0.15***	**0.45 ± 0.10**	0.45 ± 0.12	**0.55 ± 0.10***	0.51 ± 0.22	0.34 ± 0.18	0.51 ± 0.22
Hepatitis	0.45 ± 0.36	0.52 ± 0.38	0.39 ± 0.18	0.44 ± 0.21	0.36 ± 0.21	0.47 ± 0.24	**0.52 ± 0.31***	**0.66 ± 0.36***
Paw0	0.90 ± 0.08	0.94 ± 0.05	0.27 ± 0.06	0.39 ± 0.06	0.28 ± 0.07	0.40 ± 0.06	**0.92 ± 0.07***	**0.96 ± 0.04***
Paw30	0.68 ± 0.12	0.40 ± 0.08	0.26 ± 0.07	0.00 ± 0.00	0.00 ± 0.00	0.85 ± 0.08	**0.69 ± 0.13***	**0.85 ± 0.08***
Paw50	**0.62 ± 0.05***	0.75 ± 0.06	0.25 ± 0.07	0.31 ± 0.16	0.20 ± 0.11	0.30 ± 0.15	0.59 ± 0.09	**0.79 ± 0.08***
Paw70	**0.53 ± 0.13***	0.71 ± 0.09	0.13 ± 0.06	0.12 ± 0.14	0.00 ± 0.00	0.00 ± 0.00	0.47 ± 0.15	**0.75 ± 0.15***
Pima	**0.69 ± 0.06***	**0.74 ± 0.03***	0.51 ± 0.06	0.00 ± 0.00	0.59 ± 0.08	0.67 ± 0.05	0.66 ± 0.07	0.74 ± 0.04
Poker	0.09 ± 0.19	0.20 ± 0.03	0.03 ± 0.03	0.07 ± 0.09	0.00 ± 0.00	0.00 ± 0.00	**0.21 ± 0.36***	**0.30 ± 0.48***
Subc0	0.96 ± 0.05	0.98 ± 0.11	0.31 ± 0.07	0.43 ± 0.06	0.31 ± 0.07	0.43 ± 0.06	0.96 ± 0.06	**0.98 ± 0.03***
Sub30	0.67 ± 0.10	0.82 ± 0.07	0.24 ± 0.06	0.37 ± 0.06	0.05 ± 0.10	0.07 ± 0.15	**0.70 ± 0.07***	**0.88 ± 0.05***
Subc50	**0.45 ± 0.07***	0.67 ± 0.10	0.23 ± 0.06	0.33 ± 0.11	0.00 ± 0.00	0.00 ± 0.00	0.40 ± 0.13	**0.69 ± 0.11***
Subc70	0.40 ± 0.13	0.61 ± 0.32	0.24 ± 0.07	0.32 ± 0.13	0.00 ± 0.00	0.00 ± 0.00	0.40 ± 0.14	**0.61 ± 0.13***

TABLE IV

COMPARISON OF BRAF'S PERFORMANCE AGAINST THE STANDARD RANDOM FOREST ALGORITHM, TEMPC, SMOTE, AND ADABOOST USING GMEAN AS EVALUATIVE MEASUREMENT METRIC. THE BEST OVERALL RESULTS FOR EACH DATA SET ARE MARKED BY (*)

Datasets	**RFBase**	**BRAF**	**TempC**	**AdaBoost**
Abalone	0.0000	**0.1256***	0.0000	0.0000
Breast	0.5962	0.6004	0.6281	**0.6769***
Bupa	0.6913	0.7149	**0.7377***	0.6185
Car	0.7406	0.8398	**0.8656***	0.5552
Haberman	0.5741	**0.6888***	0.5558	0.5388
Hepatitis	0.5741	**0.6783***	0.6132	0.5249
Pima	0.7553	**0.7573***	0.7517	0.6948
Poker	0.0000	**0.5982***	0.3989	0.0000
Yeast	0.7953	0.8001	**0.8408***	0.4520

As shown in the results, the k value could have an impact on the and performance of BRAF may vary from one data set to another because, different data sets may have different landscapes. In order to demonstrate this, Fig. 2 pictorially illustrates the impact of k on two data sets; Colver70 and Paw70. These data sets were selected as both are artificial data sets with two attributes each which makes it is easier to visualize. Based on the experiments in this paper, the default recommended values for our BRAF method are ($k = 10$ and $p = 0.5$). Also, the imbalance ratio between the majority and minority labels should be taken in consideration when selecting the k value, as in SMOTE and other oversampling methods, it might be useful to increase the k value with the increase in the imbalance ration to compensate the difference between the labels. Moreover, BRAF could be easily combined with almost any parameter tuning and hyperparameters optimization methods suitable for random forest, in particular, or machine learning in general [10], [34].

2) Execution Time Analysis: This section provides a computational cost analysis of the proposed BRAF approach

TABLE V
TESTING AND TRAINING EXECUTION TIME ANALYSIS OF BRAF IN COMPARISON WITH SMOTE AND RF. THE RESULTS ARE IN MILLISECONDS

	BRF		SMOTE		RF	
	Test	Train	Test	Train	Test	Train
clover0	64.1	3.2	60.4	3.1	51.4	3.1
clover30	75.1	3.3	67.1	3.6	55.3	3.4
clover50	77.7	3.9	71.9	3.8	59.2	3.6
clover70	69.4	4.1	75.4	4.1	61.8	3.9
paw0	47.6	2.1	48.1	2.2	39.6	2.0
paw30	70.0	4.1	63.1	3.3	50.4	3.0
paw50	82.8	3.3	66.9	3.4	53.1	3.1
paw70	85.5	4.0	70.8	3.8	56.5	3.3
subc0	71.7	2.0	43.7	2.0	34.3	1.9
subc30	81.9	3.1	63.5	3.2	50.7	3.0
subc50	85.3	3.2	69.8	3.6	54.7	3.3
subc70	88.0	4.2	74.5	4.0	59.8	3.6
Abalone	324	9.3	330	9.4	307	8.6
breast	45.9	1.7	25.3	1.4	21.3	1.4
bupa	65.1	2.4	59.1	2.2	41.5	2.1
car-good	26.3	4.0	27.6	3.8	26.4	3.9
Haberman	33.5	2.2	34.6	2.0	27	1.9
Hepatitis	40.8	1.0	33.1	0.8	27.2	1.0
Pima	176.5	4.9	167.8	5.1	115.3	4.7
Poker	144.6	6.3	153.2	6.9	132.7	6.2

in comparison with SMOTE and RF as given in Table V. The results are in milliseconds and based on "system time" which is not very accurate but gives a good indication of the execution time. Also, the results are the average of 100 runs to compensate the inaccuracy of the method. A number of neighbors k or 1 are used for both BRAF and SMOTE. BRAF training time is similar to SMOTE but slightly higher than RF as expected. With respect to the testing/operational time which is the most important factor as training is done offline, BRAF and RF testing times are almost identical as given in Table V, this is because BRAF generates a set forest of random trees as in RF and SMOTE.

3) Statistical Significance: Paired t-test is one of the common ways to show that the superiority of one set of results over the second set against a set of data set is nonrandom. However, in the context of classification, Demsar [35] shows the t-test suffers from a few weaknesses; the main point is related to the size of the sample, unless the two classifiers are compared on a large number of data sets or in dependant runs (30 data sets or more), the t-test is valid only if the differences between the two compared results are distributed normally. Therefore, assumptions of the paired t-test are not met in this paper. It is important to mention here, repeating 10-cross validation three times on the same data set does not satisfy the t-test condition as the runs should be independent.

Demsar [35] suggests the Wilcoxon signed-rank test as a suitable alternative to the paired t-test if the t-test conditions are not satisfied. Wilcoxon signed-rank test is a nonparametric test, hence does not require the difference in the results to be normally distributed. The test is based on the rank of the difference in the performance between the two classifiers on each data set. Table VI shows the results of the Wilcoxon signed-rank test against the RF the base classifier. The results show that BRAF with k= 1, 10, and 20 FM performance (reported in Table II) is significantly better than the RF with

TABLE VI
RESULTS OF THE WILCOXON SIGNED-RANK TEST FOR BRAF WITH k = 1, 10, AND 420 AGAINST THE RF THE BASE CLASSIFIER FOR EACH MEASURE (FM AND GM). P-VALE <0.05 MEANS THAT THE BRAF RESULTS (GIVEN IN TABLE II) ARE SIGNIFICANTLY BETTER THAN RF

		BRAF1	BRAF10	BRAF20
FM	W-value	15	10.5	11.5
	Sum of neg. ranks	15	10.5	11.5
	Sum of pos. ranks	121	125.5	93.5
	Z-value	-2.7406	-2.9733	2.5738
	p-value	**0.00614**	**0.00298**	**0.01016**
GM	W-value	67	0	27
	Sum of neg. ranks	69	0	27
	Sum of pos. ranks	67	171	93
	Z-value	-0.0517	-3.7236	-1.8743
	p-value	0.96012	**0.0002**	0.06148

p-value much smaller than 0.05. Regarding the GM measure, the Wilcoxon test shows that only the BRAF with $k = 10$ is significantly better than the RF.

4) Diversity Analysis: Diversity is one of the most important properties of a classifier ensemble [36]. Diversity has been used to explain the success of several ensemble methods, both in theory and in practice [28]. In addition, Wang and Yao [37] analyzed the impact of diversity on ensemble performance against the class imbalance problem.

Diversity, in its basic form, is the degree to which the classifiers of each ensemble take different decisions on the same set of instances. Several methods for measuring diversity between base classifiers have been proposed [38]. One of the widely used methods is the disagreement measure [32] which was employed in this paper. It measures the diversity between individual decision trees based on the assertion that two decision trees are divergent when trained on same training set. Given two decision trees c_i and c_j, let $n(a, b)$ be the size of the training set on which the classification decision of c_i and c_j is a and b, respectively, e.g., $n(1, 1)$ means the number of classifiers that agreed on the same label 1. The diversity between the two decision trees is measured by

$$\text{dis}_{j,k} = \frac{n(1,-1) + n(-1,1)}{n(1,1) + n(-1,1) + n(1,-1) + n(-1,-1)}. \quad (1)$$

Diversity between the entire set of the decision trees is then determined by taking the average of the pairs of the decision trees

$$\text{dis} = \frac{2}{L(L-1)} \sum_{j=1}^{L} \sum_{k=j+1}^{L} \text{dis}_{j,k}. \quad (2)$$

Since for any two of the decision trees

$$n(1,1) + n(1,-1) + n(-1,1) + n(-1,-1) = N \quad (3)$$

we can derive

$$\text{dis} = \frac{2}{L(L-1)} \sum_{j=1}^{L} \sum_{k=j+1}^{L} n_{j,k}\,(-1,-1). \quad (4)$$

Random forest implicitly enforces diversity by employing the random subspace algorithm to deduce splitting features from the attribute space for each root node of the individual

TABLE VII

ACCENTUATES BRAF'S IMPROVEMENT IN DIVERSITY BETWEEN TREES WHEN COMPARED AGAINST RANDOM FOREST BASE CLASSIFIER, $p = 0.5$. THE BEST OVERALL RESULTS FOR EACH DATA SET ARE MARKED BY (*)

	RF	BRAF
Breast	0.297	**0.323**
Bupa	0.377	**0.420**
Car	0.059	**0.090**
Clover0	0.109	**0.281**
Clover30	0.135	**0.374**
Clover50	0.124	**0.345**
clover70	0.144	**0.362**
Haberman	0.284	**0.427**
Hepatitis	0.188	**0.376**
Pima	0.308	**0.388**
Poker	0.027	**0.343**
paw0	0.051	**0.191**
Paw30	0.104	**0.369**
paw50	0.096	**0.355**
Paw70	0.115	**0.406**
Subc0	0.046	**0.220**
Subc30	0.108	**0.328**
Subc50	0.113	**0.314**
Subc70	0.120	**0.368**

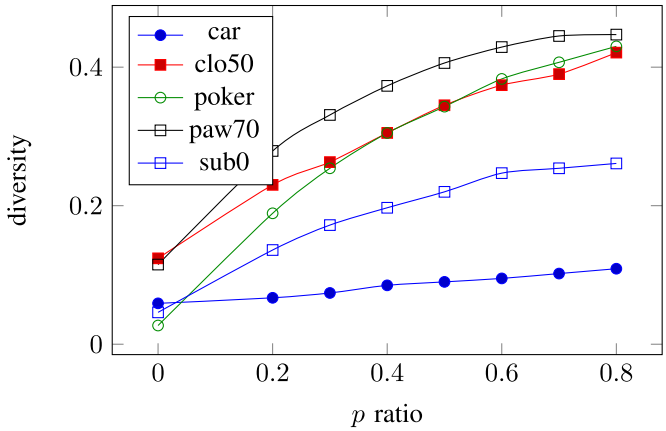

Fig. 3. BRAF's diversity, the y access is diversity of the assemble, the x excess shows the p ratio, $p = 0$ is equivalent to the standard RF, while ratio $p = 0.5$ means half of the trees in the assemble are generated form the *difficult areas* T_c.

trees [39]. BRAF boosts the already existing implicit diversity by externally generating more bootstrap samples from the *difficult areas* data set, T_c. It could be logically ascertained that the bootstrap samples generated from T_c would be different from those generated from the full data set T thereby creating diversity among the individual learning trees.

The diversity between the individual trees was measured using *disagreement measure*. Table VII shows the result of BRAF compared against the random forest base classifier. 19 data sets were used; 7 real-world application and 12 artificial data sets. As could be deduced from Table VII, BRAF enforced higher diversity across all data sets. Also, it was noted that there was BRAF's increase in diversity as p (the parameter for defining the number of trees learning from the *difficult areas*, T_c) increases. An extension of this observation is pictorially illustrated in Fig. 3 shows BRAF's diversity, the y access is diversity of the assemble, the x excess shows the p ratio, $p = 0$ is equivalent to the standard RF, while ratio $p = 0.5$ means half of the trees in the ensemble are generated form the *difficult areas* T_c.

V. CONCLUSION

Class imbalance is one of the main data challenges in classification, sampling (i.e., undersampling and oversampling) is one of the widely used methods which is applied on the data level. In this paper, we presented a novel, simple and effective approach for dealing with class imbalance. Rather than increasing the minority instances in the data set, the algorithm proposed in this paper aims to "oversample the classification ensemble" by increasing the number of classifiers that represent the minority class in the ensemble. The proposed approach could be combined with any ensemble classification method. In this paper, we adopted the random forest as it is one of the most successful and widely used classification ensemble methods. The proposed BRAF algorithm uses the nearest neighbor algorithm, which is employed by the BRAF algorithm to identify the *difficult/critical areas*. Then, a standard set of random forest trees is fed with more trees generated from *difficult/critical areas* only.

In our future work, we will investigate the relationship between the number of neighbors k and the characteristics/structure of the data set. We will look at applying BRAF concept on other ensemble classification methods. In the future, we also aim to investigate to use optimization methods to automatically discover the best performing parameters. Also, we intend to employ a genetic algorithm aimed at optimizing the best k value.

REFERENCES

[1] 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, C, Appl. Rev.*, vol. 42, no. 4, pp. 463–484, Jul. 2012.

[2] A. Awad, M. Bader-El-Den, J. McNicholas, and J. Briggs, "Early hospital mortality prediction of intensive care unit patients using an ensemble learning approach," *Int. J. Med. Informat.*, vol. 108, pp. 185–195, Dec. 2017, doi: 10.1016/j.ijmedinf.2017.10.002.

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

[4] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Inf. Sci.*, vol. 250, pp. 113–141, Nov. 2013.

[5] B. W. Yap, R. A. Rani, H. A. Rahman, S. Fong, Z. Khairudin, and N. N. Abdullah, "An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets," in *Proc. 1st Int. Conf. Adv. Data Inf. Eng. (DaEng)*. Singapore: Springer, 2014, pp. 13–22.

[6] K. Napierala, "Improving rule classifiers for imbalanced data," Ph.D dissertation, Poznan Univ. Technol., Poznań, Poland, 2013.

[7] N. Japkowicz, C. Myers, and M. Gluck, "A novelty detection approach to classification," in *Proc. IJCAI*, 1995, pp. 518–523.

[8] R. Barandela, J. S. Sánchez, V. García, and E. Rangel, "Strategies for learning in class imbalance problems," *Pattern Recognit.*, vol. 36, no. 3, pp. 849–851, 2003.

[9] M. Bader-El-Den, "Self-adaptive heterogeneous random forest," in *Proc. IEEE/ACS 11th Int. Conf. Comput. Syst. Appl. (AICCSA)*, Nov. 2014, pp. 640–646.

[10] T. Perry, M. Bader-El-Den, and S. Cooper, "Imbalanced classification using genetically optimized cost sensitive classifiers," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, May 2015, pp. 680–687.

[11] 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.

[12] H. Han, W. Y. Wang, B. H. Mao, "Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning," in *Proc. Int. Conf. Intell. Comput.* Berlin, Germany: Springer, Aug. 2005, pp. 878–887.
[13] T. Maciejewski and J. Stefanowski, "Local neighbourhood extension of smote for mining imbalanced data," in *Proc. IEEE Symp. Comput. Intell. Data Mining (CIDM)*, Apr. 2011, pp. 104–111.
[14] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, "Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*. Berlin, Germany: Springer, Apr. 2009, pp. 475–482.
[15] 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. (IJCNN), (IEEE World Congr. Comput. Intell.)*, Jun. 2008, pp. 1322–1328.
[16] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. SMC-2, no. 3, pp. 408–421, Jul. 1972.
[17] J. Laurikkala, "Improving identification of difficult small classes by balancing class distribution," in *Proc. Conf. Artif. Intell. Med. Eur.* Berlin, Germany: Springer, Jul. 2001, pp. 63–66.
[18] J. R. Quinlan, "Bagging, boosting, and C4.S," in *Proc. AAAI/IAAI*, vol. 1, 1996, pp. 725–730.
[19] R. E. Schapire and Y. Freund, *Boosting: Foundations and Algorithms*. Cambridge, MA, USA: MIT Press, 2012.
[20] J. Zhu, H. Zou, S. Rosset, and T. Hastie, "Multi-class AdaBoost," *Statist. Interface*, vol. 2, no. 3, pp. 349–360, 2009.
[21] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[22] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.
[23] C. Chen, A. Liaw, and L. Breiman, "Using random forest to learn imbalanced data," Univ. California, Berkeley, Berkeley, CA, USA, 2004, vol. 110, pp. 1–12.
[24] S. Wang and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," in *Proc. IEEE Symp. Comput. Intell. Data Mining (CIDM)*, Mar./Apr. 2009, pp. 324–331.
[25] E. Y. Chang, B. Li, G. Wu, and K. Goh, "Statistical learning for effective visual information retrieval," in *Proc. ICIP*, 2003, pp. 609–612.
[26] D. Tao, X. Tang, X. Li, and X. Wu, "Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 7, pp. 1088–1099, Jul. 2006.
[27] S. Hido, H. Kashima, and Y. Takahashi, "Roughly balanced bagging for imbalanced data," *Stat. Anal. Data Mining*, vol. 2, nos. 5–6, pp. 412–426, 2009.
[28] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits Syst. Mag.*, vol. 6, no. 3, pp. 21–45, Sep. 2006.
[29] J. Alcalá, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft Comput.*, vol. 17, nos. 2–3, pp. 255–287, 2010.
[30] D. Dua and E. K. Taniskidou, "UCI machine learning repository," Ph.D. dissertation, School Inf. Comput. Sci., Univ. California, Irvine, Irvine, CA, USA, 2017. [Online]. Available: http://archive.ics.uci.edu/ml
[31] J. Stefanowski, "Overlapping, rare examples and class decomposition in learning classifiers from imbalanced data," in *Emerging Paradigms in Machine Learning*. Berlin, Germany: Springer, 2013, pp. 277–306.
[32] D. B. Skalak, "The sources of increased accuracy for two proposed boosting algorithms," in *Proc. Amer. Assoc. Artif. Intell., AAAI, Integrating Multiple Learned Models Workshop*, vol. 1129, Aug. 1996, p. 1133.
[33] M. Bader-El-Den, E. Teitei, and M. Adda, "Hierarchical classification for dealing with the class imbalance problem," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2016, pp. 3584–3591.
[34] H. Hoos and K. Leyton-Brown, "An efficient approach for assessing hyperparameter importance," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 754–762.
[35] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.
[36] L. Kuncheva, M. Skurichina, and R. P. Duin, "An experimental study on diversity for bagging and boosting with linear classifiers," *Inf. Fusion*, vol. 3, no. 4, pp. 245–258, 2002.
[37] S. Wang and X. Yao, "Relationships between diversity of classification ensembles and single-class performance measures," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 1, pp. 206–219, Jan. 2013.
[38] E. K. Tang, P. N. Suganthan, and X. Yao, "An analysis of diversity measures," *Mach. Learn.*, vol. 65, no. 1, pp. 247–271, 2006.
[39] K. Fawagreh, M. M. Gaber, and E. Elyan, "Diversified random forests using random subspaces," in *Proc. Int. Conf. Intell. Data Eng. Automated Learn.* Cham, Switzerland: Springer, Sep. 2014, pp. 85–92.

Mohammed Bader-El-Den received the B.S. and M.Sc. degrees in computer engineering from the Arab Academy for Science and Technology, Alexandria, Egypt, in 2000 and 2003, respectively, and the Ph.D. degree in computer science from the University of Essex, Colchester, U.K., in 2009.

He was an Intern with Microsoft Research Cambridge, Cambridge, U.K., during his Ph.D. degree. He was a Visual C++ Developer with Harf for information technology, Cairo, Egypt. He was also a Sun Certified Programmer and Oracle Database Administration Certified (OCP-DBA). He was a Research Associate with Loughborough University, Loughborough, U.K. He is currently a Senior Lecturer with the University of Portsmouth, Portsmouth, U.K., where he is a member of the Computational Intelligence Group. His current research interests include data mining, classification, class imbalance, evolutionary computation, genetic programing, combinatorial optimization, and big data.

Eleman Teitei received the B.Sc. degree in computer science from Niger Delta University, Amassama, Nigeria, in 2007, and the M.Sc. degree in database systems from the University of Central Lancashire, Preston, U.K., in 2012, respectively. He is currently pursuing the Ph.D. degree in computing with the University of Portsmouth, Portsmouth, U.K.

He was a Lecturer with the Bayelsa State College of Arts and Science, Bayelsa, Nigeria. He is a Sun Certified Java Programmer, a Sun Certified Web Component Developer, a Certified Associate in Project Management, an Oracle Certified Administrator, and also a member of the Project Management Institute, USA. His current research interests include data mining and machine learning, especially tackling the class imbalance problem in supervised learning.

Todd Perry received the B.Sc. degree in computer science from the School of Computing, University of Portsmouth, Portsmouth, U.K.

He was a Researcher with the Research and Development Department, Lockheed Martin, Havant, U.K., where he was involved in several data mining projects. He is currently a Software Engineer with Huq Industries, a market intelligence company based in London, U.K. His current research interests include machine learning, big data classification, class imbalance, evolutionary computation, and combinatorial optimization.