AB-SMOTE: An Affinitive Borderline SMOTE Approach for Imbalanced Data Binary Classification

Hisham Al Majzoub and Islam Elgedawy

***Abstract*—SMOTE is an oversampling approach previously proposed to solve the imbalanced data binary classification problem. SMOTE managed to improve the classification accuracy, however it needs to generate large number of synthetic instances, which is not efficient in terms of memory and time. To overcome such drawbacks, the Borderline-SMOTE (BSMOTE) is previously proposed to minimize the number of generated synthetic instances by generating such instances based on the borderline between the majority and minority classes. Unfortunately, BSMOTE could not provide big savings regarding the number of generated instances, trading to the classification accuracy. To improve BSMOTE accuracy, this paper proposes an Affinitive Borderline SMOTE (AB-SMOTE) that leverages the BSMOTE, and improves the quality of the generated synthetic data by taking into consideration the affinity of the borderline instances. Experiments' results show the AB-SOMTE, when compared with BSMOTE, managed to produce the most accurate results in the majority of the test cases adopted in our study.**

***Index Terms*—Affinitive B-SMOTE, borderline-SOMTE, imbalanced data oversampling, SMOTE.**

I. Introduction

Imbalanced data binary classification is a very well-known problem, in which the datasets have two classes, one of them is called majority or negative class that has much more instances than the other class, which is called minority or positive class. Having imbalanced data leads to a bias towards the majority class during the classification process, which in turns leads to inaccurate classification results. This is a common problem that we can find in many different fields such as categorization [1], medicine [2], customer churn prediction [3], wine quality [4] and others, which have high imbalanced distribution of instances within the classes. In most cases, the minority class is often the intended class to be predicted, meaning that we need the classifier to generate a model that can correctly classify new data that belongs to the minority class.

Different approaches have been proposed to solve this problem, trying to minimize the imbalanced ratio such as works in [5]-[10]. Such approaches could be classified as under-sampling and oversampling approaches. In the under-sampling approaches such as works in [1], and [10] some of the majority class instances are randomly deleted to balance the numbers in the minority class. However, such approaches may lead to inaccurate results as they may delete important information needed to generate the classification model. In the other hand, the oversampling approaches such as the works in [6]-[9], they generate new instances into the minority class to balance the data. Such oversampling approach may improve accuracy, however they also have some drawbacks, such as overfitting or duplicating the data, where they won't give crucial new information for model building [5]. Hence, ensuring generating high quality non-duplicate synthetic data is crucial for the success of the oversampling approach. This is done via many heuristic strategies for data generation.

Synthetic Minority Over-sampling Technique (SMOTE) [6] is one of the popular oversampling techniques where it randomly generates new synthetic instances between the minority instances without replicating them, thus eliminating data overfitting side effect. SMOTE managed to improve the classification accuracy, however it needs to generate large number of synthetic instances (e.g. up to 500% of the minority class) [10], which is not efficient in terms of memory and time. To overcome such drawbacks, the BSMOTE [9] is previously proposed to minimize the number of generated synthetic instances by generating such instances based on the borderline between the majority and minority classes. It generates synthetic instances using borderline instances and minority class instances, as shown in Section II. Unfortunately, the BSMOTE could not provide huge savings regarding the number of generated instances, trading to the classification accuracy as shown in Section IV.

The reason for such loss in accuracy is that BSMOTE generate instances around the nearest neighbors of the borderline, and not focused inside it. This might confuse the classifier by increasing the vagueness of the borderline by increasing its boundaries, as shown in Section IV. Hence, we argue in this paper that we could have better results if we focused the oversampling inside the boundaries of the borderline and increasing its density. This motivates us to further investigate the oversampling process.

To improve BSMOTE accuracy, this paper proposes an Affinitive Borderline SMOTE (AB-SMOTE) that takes into consideration the affinity of the borderline instances to help the classifier to be more accurate in differentiating between the classes.

Experiments' results show that the AB-SOMTE, when compared with BSMOTE, managed to produce the most accurate results in the majority of the test cases adopted in the study. However, the savings in the number of generated instances were still small. Hence, we will focus on reducing the number of generated instances in future work.

This paper structure is organized as follows. Section II will

Manuscript received May 15, 2019; revised December 11, 2019.

Hisham Al Majzoub is with the Management Information Systems Department, School of Applied Sciences, Cyprus International University Nicosia, via Mersin 10 – Turkey (e-mail: hisham.m@hotmail.it).

Islam Elgedawy is with Computer Engineering Department, Middle East Technical University, Northern Cyprus Campus, 99738, Kalkanlı, Guzelyurt, Mersin 10, Turkey (e-mail: elgedawy@metu.edu.tr).

doi: 10.18178/ijmlc.2020.10.1.894

contain a brief introduction about SMOTE and Borderline SMOTE. Section III will discuss the proposed AB-SMOTE. Section IV presents the datasets used and the raw results from our experiments. In Section V, we will analyze the results of our new method against BSMOTE. Finally, in section VI, we discuss the conclusion and future work of this research.

II. Background

A. SMOTE: Synthetic Minority Over-Sampling Technique

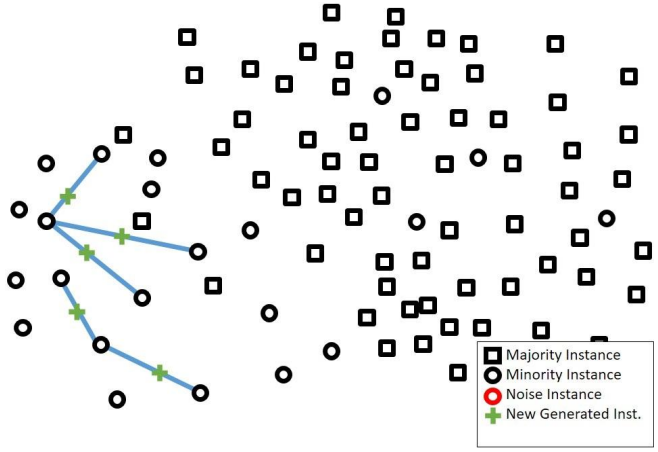

Fig. 1. SMOTE diagram.

The most known oversampling technique used in machine learning is SMOTE [6]. It checks the nearest neighbors between the instances within the minority class. The algorithm gets the percentage of new instances to be created and the number of the *K* nearest neighbors (Knn) to base its calculation on, as input from the user. The number of nearest neighbors is 5 by default. SMOTE chooses one minority instance, then calculates its Knn. After that it randomly chooses one of Knn to calculate a distance vector that is also considered as difference between the initial minority instance and the other selected instance from the same class. After the distance is calculated it will be multiplied with a random number called gap, which has a value between 0 and 1 to generate new instances falling in the line space between the selected instances. Then SMOTE continues to do the same process with other minority instances so that it will double the minority class instance number according to the percentage given by the user (e.g. 100% is the default value). Fig. 1 demonstrates the first 5 loops of the algorithm, where 5 new instances belonging to the positive minority class are generated. This is done using the following function to calculate the new values for the newly generated instances.

$$\text{New instance} = P_i + \text{gap} * (\text{distance}(P_i, P_j))$$

While Fig. 2 shows the steps that SMOTE algorithm follows to generate new instances, below are some definitions that are used in Fig. 2.

Definitions:

- Minority instance= P_i (P_1, P_2,…P_{num})
- Majority instance= N_i (N_1, N_2,…N_{num})
- *K* nearest neighbors = *Knn*
- Difference between two instances = *Dif*

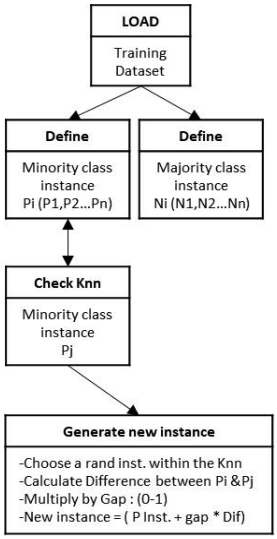

Fig. 2. SMOTE system.

SMOTE in this way will increase the number of minority instances randomly, without focusing on specific instances of the minority class. This lead to overfitting the minority class. For this reason there were other variations of the algorithm such as Safe-level-SMOTE [7], SMOTEBOOST [8], Borderline SMOTE [9] and others, to overcome this problem.

B. BSMOTE: Borderline-SMOTE

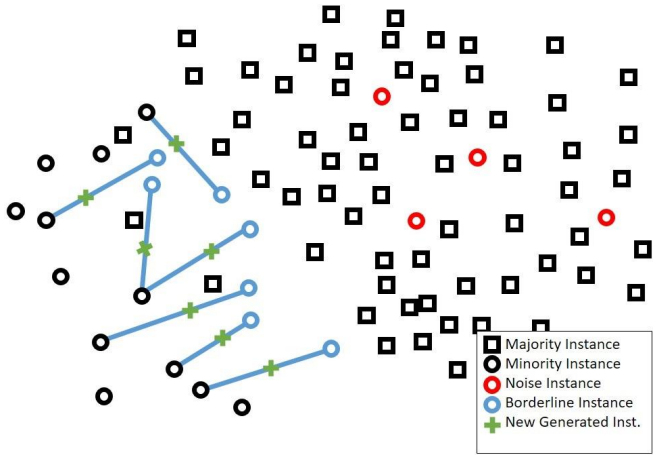

Fig. 3. Borderline SMOTE.

BSMOTE proposed in [9] eliminates some instances from the computation function considering them as noise, or safe instances. It focuses its computation on the borderline instances that fall between the two classes, shown in blue color in Fig. 3 to generate new instances.

The borderline instances are chosen by calculating the number of majority instances (*M'*) that are found within the *M* nearest neighbors (*Mnn*) between each instance belonging to the minority class and all other instances within the dataset. Such that, if (*M'*) value is between *M*/2 and M, the minority instance (*Pi*) is considered to be as a borderline instance (P'_i). After creating the new subset that have all the minority borderline instances, BSMOTE measures *Knn* between borderline instances and other minority instances, then generates the new instance using the following function:

$$\text{New instance} = P'_i + \text{gap} * (\text{distance}(P'_i, P_j))$$

where P'_i is the borderline minority instance, P_j is the random

chosen *Knn* minority instance, and gap is a random number having value between 0 and 1. Fig. 4 demonstrates how BSMOTE works.

Definitions:

- *M* nearest neighbors= *Mnn*
- Borderline Instance= P'_i (P'_1, P'_2…P'_{num})

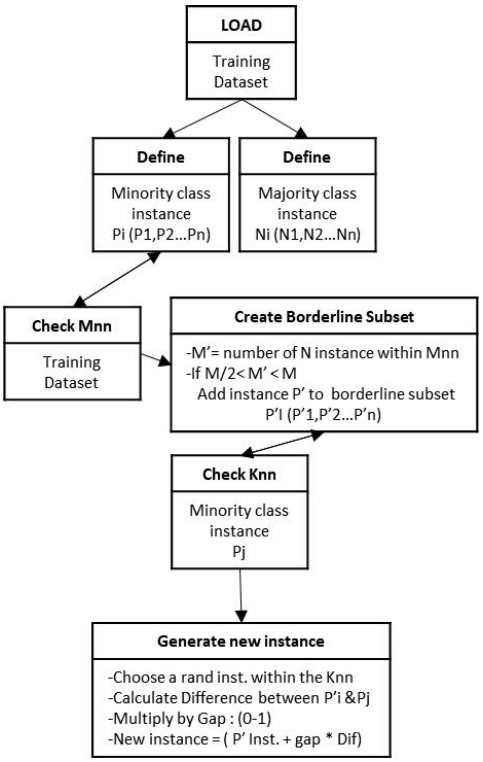

Fig. 4. BSMOTE approach.

There is also another version of BSMOTE that is called BSMOTE2 where the algorithm can take points from the majority class. But using an instance from the majority instance, the gap values used will be between 0 and 0.5 so that the new created instance can be adjacent to the borderline. We can see that in BSMOTE creation of new instances is somehow focused around the nearest neighbors of the borderline, but not the borderline itself. We would have better results if we focused on the existing boundaries of the borderline to increase its density. We denote such oversampling strategy as "borderline affinity". Hence, we propose the AB-SMOTE to adopt the borderline affinity oversampling strategy.

III. AB-SMOTE: Affinitive Borderline SMOTE

After researching SMOTE and BSMOTE, we noticed that we could develop BSMOTE in a way that we can make the generation of new instances computed only between borderline instances, thus increasing the minority instances within that area. We named our approach Affinitive Borderline SMOTE (AB-SMOTE) that works very similar to the BSMOTE but instead of checking Knn between the borderline instances and all minority instances, it only checks Knn within the borderline instances. Thus excluding the noise and/or safe instances that were used before to generate new instances. Fig. 5 shows AB-SMOTE diagram uses only instances within the borderline area to generate new instances. Whereas Fig. 6 describes how AB-SMOTE works. AB-SMOTE defines the minority and majority class instances, then computes for every minority instance it's M nearest neighbors within the whole training data. At each iteration, it counts the number of majority instances M' found within the M nearest neighbors, and if this M' number falls between M/2 and M, the intended minority instance is considered as a borderline instance and copied to a new subset having a name borderline (danger). In case M' was less than M/2 the minority instance is treated as a safe one, where most of its surroundings are from the minority class, and if it was equal to M, it is considered as a noise because all of the M nearest neighbors are from the majority class. Later the algorithm checks Knn within the borderline instances, randomly chooses one of the Knn, calculate the distance between the two instances and apply the following function to compute the value of the new instance. Which makes the creation of new instances focused on the borderline.

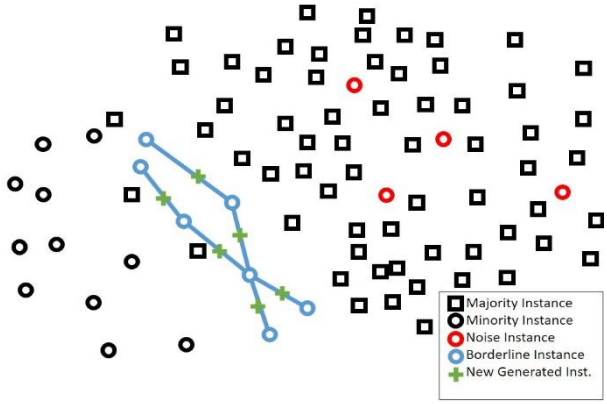

Fig. 5. Affinitive borderline SMOTE.

$$\text{New instance} = P'_i + \text{gap} * (\text{distance}(P'_i, P'_j))$$

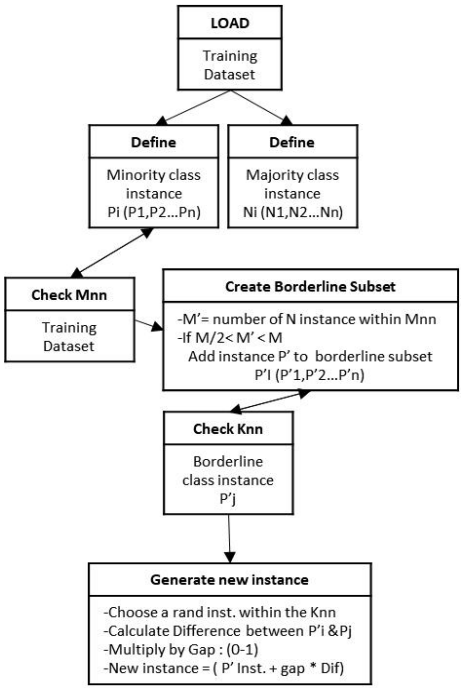

Fig. 6. Affinitive borderline SMOTE approach.

IV. Evaluation

In order to do the evaluation for our new approach, we

used WEKA [11] software for machine learning, which is an open source java script program. That allowed us to modify its codes. We also used different supervised datasets shown in Table I obtained from different online repositories. These real life datasets are widely used in machine learning scientific research. These datasets were downloaded from Crowd analytics, UC Irvine Machine Learning Repository, Keel, and IBM. We adopted the imbalanced threshold considered in [12] in choosing the datasets. That if minority number of instances are less than 40% of the number of instances of the majority class, the dataset is considered unbalanced and chosen for our evaluation.

TABLE I: DATASETS USED IN OUR RESEARCH

Datasets	Total	Minority	Majority	IR Ratio	Source
Abalone9-18	731	42	689	6%	[13]
Customer Churn Wireless Telecom	5000	707	4293	17%	[14]
Telco Customer Churn	7043	1869	5174	36%	[15]
Haberman	306	81	225	36%	[16]
Employe Attrition	1470	237	1233	19%	[17]
Flare	1109	43	1066	4%	[18]
Wine Quality	1493	53	1546	3.4%	[4]
Sales: Win Loss	78025	17627	60398	29.18%	[19]

We classified our datasets using decision tree classification algorithm, which is called J48 in WEKA. The classification stage was conducting using 5-folds cross-validation to have enough positive minority class instances in every fold to minimize the data distribution problems [20], [21]. As we are dealing with unbalanced data set, evaluating the classifier with classification accuracy alone do not give a good overview about the classifier accuracy in predicting the minority class, instead a confusion matrix such as Table II is used for checking different metrics to get a good overview about the classifier prediction power [22]. The majority class is considered as Negative, while the minority class is Positive. Using the confusion matrix, true positive (TP) number should be as high as it might get within the limit of total number of the real positive value. Which implies that the model is correctly predicting the instances that have an actual positive class to be in a positive class.

TABLE II: CONFUSION MATRIX

		Actual Values: Positive	Actual Values: Negative		
Predicted Values	Positive	TP	FP	P.P.V.	$\frac{TP}{(TP+FP)}$
	Negative	FN	TN	N.P.V	$\frac{TN}{(FN+TN)}$
		Recall: $\frac{TP}{(TP+FN)}$	Specificity: $\frac{TN}{(TN+TP)}$	Accuracy: $\frac{(TP+TN)}{(TP+FP+TN+FN)}$	

We evaluated the classifiers with recall and f-measures values that show up as an output in WEKA after applying the decision tree classifier with 5-folds cross-validation.

1. Recall: which is the true positive rate where the classifier had correctly classified an actual minority instance to be in the minority class
2. F-measure: which is the harmonic mean of recall and precision [22], that can be measured with these functions

F-measure= 2*(Precision*Recall)/(Precision+ Recall)

The best values for recall and f-measure is when they tend to reach the value of one. Because we are working with supervised datasets, the instances have a known class value, and when applying 5- fold cross-validation means, that we split the data into 5 folds, 4 for training and one for testing, and it is done 5 times, and each time the testing portion is different than the other. WEKA gets the results from each fold, compute and output its average inside WEKA window. When a model is created using a training subset, it is applied on a test subset and compared the class value between the actual instances and predicted one to create the results. The results of the original data sets are obtained and recorded, then we applied different oversampling techniques such as SMOTE, BSMOTE, and AB-SMOTE using different tuning options that are shown in Table III. Where Mnn stands for the M number of the nearest neighbors between minority and whole dataset to create the borderline subset, Knn stands for the number for K nearest neighbors, G stands for Gap, the number which is multiplied by the distance to create the new instance, percentage is the value that calculate the number of new generated instances, P' is the borderline instances, and P is the minority instances.

TABLE III: DIFFERENT METHODS USED IN OUR EXPERIMENT

Algorithm	**Mnn**	**Knn**	**GAP**	**Percentage**
1.0: Original	-	-	-	-
2.0: SMOTE	-	5	0 - 1	100
3.0: BSMOTE P' & P	5	5	0 - 1	100
3.1: BSMOTE P' & P	5	8	0 - 1	100
3.2: BSMOTE P' & P	8	5	0 - 1	100
3.3: BSMOTE P' & P	8	8	0 - 1	100
3.4: BSMOTE P' & N	5	5	0 - 0.5	100
4.0: AB-SMOTE P' & P'	5	5	0 - 1	100
4.1: AB-SMOTE P' & P'	5	8	0 - 1	100
4.2: AB-SMOTE P' & P'	8	5	0 - 1	100
4.3: AB-SMOTE P' & P'	8	8	0 - 1	100

Table IV to Table XI show the obtained recall and f-measure values. Gen resembles the number of generated new instances. We bolded the best obtained values.

TABLE IV: RESULTS FROM ABALONE DATASET

Majority= 689; Minority = 42	I.R.= 6.1%		
Oversampling Technique	Gen	Recall	F-measure
1.0: Original	0	0.167	0.241
2.0: SMOTE *k*=5	42	0.381	0.474
3.0: BSMOTE P' & P: *M*=5 *k*=5	40	0.415	0.507
3.1: BSMOTE P' & P: *M*=5 *k*=8	40	0.415	0.511
3.2: BSMOTE P' & P: *M*=8, *k*=5	39	0.469	0.524
3.3: BSMOTE P' & P: *M*=8, *k*=8	39	0.395	0.496
3.4: BSMOTE P' & N: *M*=5 *k*=5 G[0-0.5]	40	0.378	0.434
4.0: AB-SMOTE P' & P': *M*=5 *k*=5	40	0.427	0.493
4.1: AB-SMOTE P' & P': *M*=5 *k*=8	40	0.439	0.529
4.2: AB-SMOTE P' & P': *M*=8 *k*=5	39	0.432	0.519
4.3: AB-SMOTE P' & P': *M*=8 *K*=8	39	**0.494**	**0.559**

As per Table IV, the best algorithm was 4.3 AB-SMOTE with Mnn=8 and Knn=8 where we got the highest Recall and F-measure, even outperforming the regular SMOTE.

As per Table V, the 4.1 AB-SMOTE with Mnn=5 and Knn= 8 outperformed the other approaches with better Recall and F-measure. But with this particular dataset the original dataset without any oversampling got the highest

classification accuracy for the minority class.

TABLE V: RESULTS FROM CUSTOMER CHURN DATASET

Majority= 4293; Minority = 707	I.R.= 16.47%		
Oversampling Technique	Gen	Recall	F-measure
1.0: Original	0	0.636	0.748
2.0: SMOTE *k*=5	707	0.631	0.749
3.0: BSMOTE P' & P: *M*=5 *k*=5	534	0.573	0.708
3.1: BSMOTE P' & P: *M*=5 *k*=8	534	0.597	0.718
3.2: BSMOTE P' & P: *M*=8, *k*=5	517	0.592	0.719
3.3: BSMOTE P' & P: *M*=8, *k*=8	517	0.576	0.709
3.4: BSMOTE P' & N: *M*=5 *k*=5 G[0-0.5]	534	0.502	0.647
4.0: AB-SMOTE P' & P': *M*=5 *k*=5	534	0.578	0.704
4.1: AB-SMOTE P' & P': *M*=5 *k*=8	534	**0.602**	**0.72**
4.2: AB-SMOTE P' & P': *M*=8 *k*=5	517	0.565	0.698
4.3: AB-SMOTE P' & P': *M*=8 *K*=8	517	0.565	0.698

TABLE VI: RESULTS FROM HARBERMAN DATASET

Majority = 225 ; Minority = 81	I.R.= 36.00%		
Oversampling Technique	Gen	Recall	F-measure
1.0: Original	0	0.272	0.331
2.0: SMOTE *k*=5	81	0.617	0.643
3.0: BSMOTE P' & P: *M*=5 *k*=5	65	0.534	0.576
3.1: BSMOTE P' & P: *M*=5 *k*=8	65	0.627	0.635
3.2: BSMOTE P' & P: *M*=8, *k*=5	61	0.676	0.613
3.3: BSMOTE P' & P: *M*=8, *k*=8	61	0.57	0.572
3.4: BSMOTE P' & N: *M*=5 *k*=5 G[0-0.5]	65	0.5	0.559
4.0: AB-SMOTE P' & P': *M*=5 *k*=5	65	0.527	0.575
4.1: AB-SMOTE P' & P': *M*=5 *k*=8	65	0.671	**0.643**
4.2: AB-SMOTE P' & P': M=8 *k*=5	61	**0.697**	0.639
4.3: AB-SMOTE P' & P': *M*=8 *K*=8	61	0.57	0.581

As per Table VI, the 4.2 AB-SMOTE having Mnn=8 and Knn=5 got the best Recall value, whereas 4.1 AB-SMOTE with Mnn=5 and Knn=8 got the best F-measure.

TABLE VII: RESULTS FROM EMPLOYEE ATTRITION DATASET

Majority =1233 ; Minority = 237	I.R.= 19.22%		
Oversampling Technique	Gen	Recall	F-measure
1.0: Original	0	0.181	0.265
2.0: SMOTE k=5	237	0.519	0.58
3.0: BSMOTE P' & P: *M*=5 *k*=5	236	0.526	0.588
3.1: BSMOTE P' & P: *M*=5 *k*=8	236	0.513	**0.871**
3.2: BSMOTE P' & P: *M*=8, *k*=5	232	**0.542**	0.598
3.3: BSMOTE P' & P: *M*=8, *k*=8	232	0.516	0.587
3.4: BSMOTE P' & N: *M*=5 *k*=5 G[0-0.5]	236	0.463	0.544
4.0: AB-SMOTE P' & P': M=5 k=5	236	0.533	0.603
4.1: AB-SMOTE P' & P': M=5 k=8	236	0.51	0.603
4.2: AB-SMOTE P' & P': M=8 k=5	232	0.529	0.595
4.3: AB-SMOTE P' & P': M=8 K=8	232	0.529	0.593

As per Table VII, the approach 3.2 BSMOTE with Mnn=8 and Knn=5 got the best Recall, while the approach 3.1 BSMOTE with Mnn=5 and Knn=8 got the best F-measure.

TABLE VIII: RESULTS FROM TELCO CUSTOMER CHURN DATASET

Majority= 5174 ; Minority=1869	I.R.= 36.12%		
Oversampling Technique	Gen	Recall	F-measure
1.0: Original	0	0.489	0.543
2.0: SMOTE *k*=5	1869	0.917	0.731
3.0: BSMOTE P' & P: *M*=5 *k*=5	1101	0.895	0.677
3.1: BSMOTE P' & P: *M*=5 *k*=8	1101	0.894	0.677
3.2: BSMOTE P' & P: *M*=8, *k*=5	949	0.889	0.664
3.3: BSMOTE P' & P: M=8, *k*=8	949	0.888	0.663
3.4: BSMOTE P' & N: *M*=5 *k*=5 G[0-0.5]	1101	0.854	0.657
4.0: AB-SMOTE P' & P': *M*=5 *k*=5	1101	0.897	0.678
4.1: AB-SMOTE P' & P': *M*=5 *k*=8	1101	**0.898**	**0.679**
4.2: AB-SMOTE P' & P': *M*=8 *k*=5	949	0.891	0.665
4.3: AB-SMOTE P' & P': *M*=8 *K*=8	1101	0.89	0.665

As per Table VIII, AB-SMOTE outperformed in Recall and F-measure while using Mnn=5 and Knn=8.

TABLE IX: RESULTS FROM FLARE DATASET

Majority = 1066; Minority = 43	I.R.= 4.03%		
Oversampling Technique	Gen	Recall	F-measure
1.0: Original	0	0	0
2.0: SMOTE *k*=5	43	0.36	0.47
3.0: BSMOTE P' & P: *M*=5 *k*=5	39	0.268	0.389
3.1: BSMOTE P' & P: *M*=5 *k*=8	39	0.268	0.37
3.2: BSMOTE P' & P: *M*=8, *k*=5	41	0.25	0.347
3.3: BSMOTE P' & P: *M*=8, *k*=8	41	**0.369**	0.434
3.4: BSMOTE P' & N: *M*=5 *k*=5 G[0-0.5]	39	0.171	0.262
4.0: AB-SMOTE P' & P': *M*=5 *k*=5	39	0.293	0.421
4.1: AB-SMOTE P' & P': *M*=5 *k*=8	39	0.341	**0.463**
4.2: AB-SMOTE P' & P': *M*=8 *k*=5	41	0.286	0.397
4.3: AB-SMOTE P' & P': *M*=8 *K*=8	41	0.25	0.359

In Table IX BSMOTE with Mnn and Knn = 8 got best recall, while AB-SMOTE with Mnn=5 and Knn=8 get the best F-measure.

TABLE X: RESULTS FROM WINE QUALITY DATASET

Majority= 1546 ; Minority= 53	I.R.= 3.43%		
Oversampling Technique	Gen	Recall	F-measure
1.0: Original	0	0	0
2.0: SMOTE *k*=5	53	0.151	0.215
3.0: BSMOTE P' & P: *M*=5 *k*=5	53	0.151	**0.215**
3.1: BSMOTE P' & P: *M*=5 *k*=8	53	**0.16**	0.205
3.2: BSMOTE P' & P: *M*=8, *k*=5	53	0.151	**0.215**
3.3: BSMOTE P' & P: *M*=8, *k*=8	53	**0.16**	0.205
3.4: BSMOTE P' & N: *M*=5 *k*=5 G[0-0.5]	53	0.132	0.193
4.0: AB-SMOTE P' & P': *M*=5 *k*=5	53	0.151	**0.215**
4.1: AB-SMOTE P' & P': *M*=5 *k*=8	53	**0.16**	0.205
4.2: AB-SMOTE P' & P': *M*=8 *k*=5	53	0.151	**0.215**
4.3: AB-SMOTE P' & P': *M*=8 *K*=8	53	**0.16**	0.205

As per Table X, we got a tie between Recall and F-measures results when oversampling with BSMOTE and AB-SMOTE

TABLE XI: RESULTS FROM TELCO SALES WIN LOSS DATASET

Majority= 60398; Minority= 17627	I.R.= 29.18%		
Oversampling Technique	Gen	Recall	F-measure
1.0: Original	0	0.644	0.695
2.0: SMOTE *k*=5	17627	0.816	0.838
3.0: BSMOTE P' & P: *M*=5 *k*=5	15176	0.799	0.817
3.1: BSMOTE P' & P: *M*=5 *k*=8	15176	0.803	0.823
3.2: BSMOTE P' & P: *M*=8, *k*=5	14320	0.787	0.81
3.3: BSMOTE P' & P: *M*=8, *k*=8	14320	0.793	0.816
3.4: BSMOTE P' & N: *M*=5 *k*=5 G[0-0.5]	15176	0.78	0.81
4.0: AB-SMOTE P' & P': *M*=5 *k*=5	15176	**0.804**	**0.826**
4.1: AB-SMOTE P' & P': *M*=5 *k*=8	15176	0.803	0.826
4.2: AB-SMOTE P' & P': *M*=8 *k*=5	14320	0.798	0.822
4.3: AB-SMOTE P' & P': *M*=8 *K*=8	14320	0.797	0.82

Finally, in Table XI the approach 4.0 AB-SMOTE with Mnn and Knn equal 5 got the best Recall and F-measure.

V. RESULTS ANALYSIS

As we can see from previous results, BSMOTE and AB-SMOTE managed to provide better results than the original SMOTE approach except in one case in Table V. This means focusing on the boundary to generate the synthetic data is a very promising strategy. Hence, we can deduce that the borderline affinity strategy is better than

randomly generating new instances from safe and/or noise instances.

For every borderline-oriented approach, every dataset will require its proper parameters tuning to get the best fitting approach for this dataset. However, to see overall performance of the borderline-oriented approaches regardless of their parameters tuning, we marked the approach with the best value of recall and F-measure from the previous results' tables and constructed Table XII.

TABLE XII: BEST RESULTS OF THE OVERSAMPLING

DATASET	Recall		F-Measure	
	BSMOTE	AB-SMOTE	BSMOTE	AB-SMOTE
Abalone		x		X
Churn		x		X
Harberman		x		X
Employee Attrition	x		X	
Telco Customer churn		x		X
Flare	x			X
Wine Quality	x	x	X	X
Sales, win loss		x		X
TOTAL	**3**	**6**	**2**	**7**

From Table XII we can notice that AB-SMOTE outperformed BSMOTE by getting the most number of best values in terms of Recall and F-measure. This means focusing on increasing the density of the borderline area is a more effective oversampling strategy rather than increasing the boundaries of the borderline as in BSMOTE. Hence, we can say that the borderline affinity oversampling strategy worked very well with the decision tree classifier. In Future work, we will study to see if borderline affinity strategy holds for other classifiers or not.

Table IV to Table XI also show that both BSMOTE, and AB-SMOTE could not significantly reduce the number of generated instances. Hence, we will investigate this issue in more depth in future work. This will be done by adopting a selective strategy that chooses certain instances in the borderline to be used in new instances generation rather than randomly choosing the instances. This will require examining the density distribution of the borderline instances, then carefully generates the new instances in a way that does not disrupt the calculated borderline density distribution.

VI. CONCLUSION

In this paper, we proposed the AB-SOMTE to improve the accuracy of the existing SMOTE and BSMOTE. This is done by taking the affinity of borderline instances into consideration when generating the synthetic data. We compared the accuracy of the three approaches against different data sets. Experiments' results show the AB-SOMTE managed to produce the most accurate results in the majority of the cases. This means the proposed borderline affinity oversampling strategy is very promising, and could be leveraged more to select fewer instances from the borderline to reduce the number of generated data.

We believe this a very important direction for future work, as both BSMOTE, and AB-SMOTE could not provide big savings in the number of generated data. By selecting fewer key borderline instances to generate new instances, we could heavily minimize the number of generated synthetic data. However, this should be carefully done without disrupting the borderline density distribution.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Al Majzoub and Elgedawy created the theory, Al Majzoub done the coding and evaluated the algorithm; Al Majzoub and Elgedawy analyzed the results; Al Majzoub wrote the paper and Elgedawy reviewed and edited it. All authors had approved the final version.

REFERENCES

[1] A. Sun, E. P. Lim, and Y. Liu, "On strategies for imbalanced text classification using SVM: A comparative study," *Decis. Support Syst.*, vol. 48, no. 1, pp. 191-201, 2009.

[2] F. B. Tek, A. G. Dempster, and I. Kale, "Parasite detection and identification for automated thin blood film malaria diagnosis," *Comput. Vis. Image Underst.*, vol. 114, no. 1, pp. 21-32, 2010.

[3] S. A. Qureshi, A. S. Rehman, A. M. Qamar, A. Kamal, and A. Rehman, "Telecommunication subscribers' churn prediction model using machine learning," in *Proc. Eighth Int. Conf. Digit. Inf. Manag.*, 2013, no. September, pp. 131-136.

[4] *Keel Datasets, Wine Quality*. [Online]. Available: https://sci2s.ugr.es/keel/dataset.php?cod=1322

[5] M. Bekkar, D. Alitouche, T. Akrouf, and T. A. Alitouche, "Imbalanced data learning approaches review," *Data Min. Knowl.,* vol. 3, no. 4, pp. 15-33, 2013.

[6] 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, 2002.

[7] C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap, "Safe-level-SMOTE: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem," *Lect. Notes Comput. Sci.*, vol. 5476, pp. 475-482, 2009.

[8] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," in *Proc. the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases*, 2003, pp. 107-119.

[9] H. Han, W. Wang, and B. Mao, "Borderline-SMOTE : A new over-sampling method in imbalanced data sets learning," in *Proc. the International Conference on Intelligent Computing*, 2005, pp. 878-887.

[10] M. Bach, A. Werner, J. Żywiec, and W. Pluskiewicz, "The study of under- and over-sampling methods' utility in analysis of highly imbalanced data on osteoporosis," *Inf. Sci. (Ny).*, vol. 384, pp. 174-190, 2017.

[11] WEKA. [Online]. Available: https://www.cs.waikato.ac.nz/ml/weka/index.html

[12] 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," *Knowledge-Based Syst.*, vol. 42, pp. 97-110, 2013.

[13] Keel Datasets, Abalone9-18. [Online]. Available: http://sci2s.ugr.es/keel/dataset.php?cod=116

[14] Crowd Analytix. [Online]. Available: http://www.crowdanalytix.com/contests/why-customer-churn/

[15] IBM Analytics Customer Churn Dataset. [Online]. Available: https://www.ibm.com/communities/analytics/watson-analytics-blog/predictive-insights-in-the-telco-customer-churn-data-set/

[16] UC Irvine Machine Learning Repository. [Online]. Available: https://archive.ics.uci.edu/ml/datasets.html.

[17] IBM Analytics. [Online]. Available: http://www.ibm.com/communities/analytics/watson-analytics-blog/guide-to-sample-datasets/

[18] Keel Datasets, Solar Flare. [Online]. Available: https://sci2s.ugr.es/keel/dataset.php?cod=1331#sub1

[19] IBM Analytic, Win Loss. [Online]. Available: https://www.ibm.com/communities/analytics/watson-analytics-blog/guide-to-sample-datasets/

[20] J. Cervantes, F. Garcia-Lamont, L. Rodriguez, A. López, J. R. Castilla, and A. Trueba, "PSO-based method for SVM classification on skewed data sets," *Neurocomputing*, vol. 228, pp. 187-197, 2017.

[21] V. López, A. Fernández, and F. Herrera, "On the importance of the validation technique for classification with imbalanced datasets:

Addressing covariate shift when data is skewed," *Inf. Sci. (Ny).*, vol. 257, pp. 1-13, 2014.
[22] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PLoS One*, vol. 10, no. 3, pp. 1-21, 2015.

Hisham Al Majzoub received the masters of science in business administration degree from Arts, Sciences and Technology University in Lebanon in September 2011. He was working as an IT branch manager in American University of Science and Technology in Saida – Lebanon from 2001 to 2014. He is pursuing the Ph.D. degree in management information systems at Cyprus International University. His researches focus on imbalanced class problem found in datasets that are used in machine learning.

Islam Elgedawy is an associate professor at the Computer Engineering Department, Middle East Technical University-Northern Cyprus Campus. He received his B.Sc. and M.Sc. degrees in computer science from Alexandria University, Egypt in 1996, and 2000, respectively, and his Ph.D. degree in computer science from RMIT University, Australia in 2007. His work focuses on the areas of service-oriented computing, organic computing, software engineering, and big data analytics. He is an author and co-author of many publications in international journals and conferences, also he has a growing record of consultancy and professional services.