Journal of Engineering and Technology

ISSN 2180-3811 eISSN 2289-814X https://journal.utem.edu.my/index.php/jet/index

MACHINE LEARNING SHREWD APPROACH FOR AN IMBALANCED DATASET CONVERSION SAMPLES

Shahzad Ashraf[1*] Tauqeer Ahmed[1]
[1]College of Internet of Things Engineering, Hohai University, Changzhou, China.
* nfc.iet@hotmail.com

Article history:
Received Date: 2020-03-18
Accepted Date: 2020-06-02

Keywords: Classification; Machine learning; SMOTE; Spread Subsampling; Classimbalance

Abstract— The imbalance data applies to at least one of the classes, which are typically exceeded by the other ones. The Machine Learning Algorithm (Classifier) trained with an imbalance dataset predicts the majority class (frequently occurring) more than the other minority classes (rarely occurring). Training with an imbalance dataset poses challenges for classifiers; however, applying suitable techniques for reducing class imbalance issues can enhance the classifier's performance. We take a imbalanced dataset from an educational context. Initially, all shortcomings regarding the classification of the imbalanced dataset have been examined. After that, we apply data-level algorithms for class balancing and compare the performance of classifiers. The performance of the classifier is measured using the underlying information in their confusion matrices such as accuracy, precision, recall, and f-measure. It shows that classification with an imbalance dataset may produce higher accuracy but low precision and recall for the minority class. The analysis confirms that both under sampling and oversampling are useful for balancing datasets; however, oversampling dominates.

I. Introduction

There has been an enormous increase in the production of data across a wide range of fields. The dataset classification is a unique data mining technique. Its objective is to determine which target class belongs to a specific object in an unknown class. The result of a classification algorithm is generally related to data characteristics. One such algorithm, like SVM (support vector machine) [1], possessed numerous Special advantages in solving specific problems of classification, such as low sample numbers, nonlinearity, and high-dimensional pattern recognition.

Moreover, the classification exactness of the minority class is often more valuable. In the case of imbalanced data, most-class examples will have a more significant influence on the classifier, causing its classification weight to be in favor of the majority class and then seriously affecting the classification hyperplane distribution. It is very critical that classification approaches can be improved at the algorithm or data level to solve the imbalanced classification of data, which is currently a trend problem in the field of data mining research. The organizations are keen to process the collected data and pull out valuable information that can support their decision making [2]. Data Mining (DM) [3], aims to collect, organize, and process vast amounts of data to identify useful unseen patterns. Internet, being a vital tool of communication and information, is offering exclusive benefits to both educators and students. Classification is one of the significant application fields in the data mining wherein the instances (records) in a dataset are grouped in more than one class. The classification can be Pass/Fail in the pedagogical environment or classifying flowers in different types [4]. The Classifier gains knowledge from a prearranged training dataset, henceforth, to organize the instances from the unseen dataset, the class imbalance problem appears in datasets having an exceedingly unfair ratio between the classes [5]. This poses challenges for data mining and classification process. Classifiers trained with an imbalance dataset tend to predict the majority class (frequently occurring) more than the minority class (rarely occurring) [6]. It is because standard classifiers are designed to concentrate on minimizing the overall classification error regardless of the class distribution. It is harder for the classifier to learn from the class having a fewer number of instances.

Attention has been focused on the classification of imbalanced data. In recent years, many researchers have been attracted by classification algorithms based on imbalanced data. Study approaches to the classification of imbalanced data by the SVM are currently primarily divided into two categories: improvement of methods at the algorithm level and improvements at the level of data. The weighted SVM of the penalty coefficient C is used at the algorithm level to control the various costs for misclassification errors of various classes. The minority class is generally charged a higher cost of error classification, and the majority class is charged a low cost of misclassification. In addition, the AdaBoost algorithm, the integrated multi-classifier algorithm, and an enhancing kernel space-based algorithm are widely utilized. Two fundamental approaches are present at the data level: a strategy for over-sampling of the minority specimens and the undersampling of the majority specimens. The technique of over-sampling uses specific approaches to balance class distributions, such as the duplication of minority example or artificial synthesizing of new minority class examples using algorithms. In addition to oversampling, undersampling is a standard method of managing unbalanced datasets. Under-sampling balances the distribution of data classes with the elimination of majority class examples as the Tomek Links algorithm [7].

The significant contribution of this experimental research is to draw attention toward the misclassification issues, which results from training a classifier with a dataset where the instances in class are not balanced. This research clarifies that higher accuracy may not be enough to rank classifiers. This work proposes that classifiers' performance can be enhanced with the implementation of sampling algorithms for eradicating the class imbalance problem. To spotlight, we consider a dataset from an educational institute where the majority of the attributes have real values [8].

In this study, we extracted underlying information from the confusion matrix and compared the classifier's performance for the majority and minority classes. This analysis makes it evident that accuracy may not appear as rigid evaluation criteria; instead, the focus should be on classifier performance for minority and majority classes

II. RELATED WORK

Numerous solutions are proposed to do away with the class imbalance problem. They are either at the data level or algorithm level. At the data level, the proposed algorithms use various forms of re-sampling techniques such as undersampling and oversampling. At the algorithmic level, solutions include cost-sensitive learning, fine-tuning of the probabilistic estimation at the tree leaf (in decision tree implementation), adjusting decision threshold, and preferring recognition-based learning rather than discrimination-based (in 2-class) learning [9].

Educational Data Mining (EDM) [10], mines significant patterns in the data, collected from a pedagogical domain, to optimize the learner and learning environment. The classification models in an educational environment forecast the learner's expected academic outcome. One such prediction model forecasts the result (grade) of the student in a specific course. Once, the model predicts the student with poor final grades, at that moment, the instructor intervenes to tutor the student and lead him/her toward achieving the improved final result. The limited number of students in a course leaves these datasets with a lower number of instances [11].
Moreover, a wide range of students' attributes, such as attendance, marks in assessment tools, CGPA, credit hours, and marks in pre-requisite courses, possess real values. The dataset in such environments suffers from class imbalance issues, wherein fewer learners have chances to perform unsatisfactorily. In this paper, we consider a small imbalanced dataset, with attributes having nominal and real value, from a course in an institute.

In an empirical study Hernandez [12] demonstrates the use of oversampling and under sampling algorithms to improve the accuracy of instance selection methods on imbalanced databases. There results in yield that both oversampling and under sampling techniques improve accuracy. To enhance the performance of classifiers based on emerging patterns, Loyola-González [13], use oversampling and under sampling methods. Similarly, E. Osmanbegovic et al. [14], implement Machine Learning algorithms to classify students into binary classes (A and B). The dataset suffers from an imbalanced ratio, and the number of instances in class 'B' is much bigger than that of class 'A.' The results show that each of the applied algorithms has produced higher precision and recall for class B. Naïve Bayes being the better-performing classifier yields a recall of 0.500 for class 'A' and 0.851 for class 'B.' All the implemented algorithms (Naïve Bayes, Multilayer Perceptron and Decision Tree) produced higher Recall, FP rate, Precision value for B than A. Besides, D Kabakchieva [15], makes use of classification algorithms to classify students into 5 classes (excellent, very good, good, average, and bad). The dataset has over 4,000 instances form 'very good and 'good' classes and around 500 or less than that for the other 3 classes. Decision Tree (J48) achieves less than 0.100 recall values for 'average' and 'excellent' class compared to other classes that achieve nearly or more than 0.70 recall.

Some previous contributions regarding class distribution and the imbalance ratio are being presented in Table 1. Similarly, the difference in the performance evaluation of 7 classes ranges from 0 to 83% in the result is shown by [16]. Some results are evidence of high diversity between the F-Measure of majority and minority classes. The Multilayer Perceptron has achieved the highest accuracy of 75%, but on the other hand, the difference between F-Measure of majority and minority classes is 0.244 (nearly one fourth); similarly, it is almost 50% in case of SMO. This draws attention toward the need for proper class distribution before performing experiments to achieve reasonable results for all the considered classes.

Table 1. Comparison of class balancing ratio of various works

Work from authors	Class Distribution/Imbalance Ratio					
E. Osmanbegovic et al. [15]	**Class**	A			B	
	Instances	62			195	
	Imbalanced Ratio	1			3.14	
R. Asif et al. [16] (Dataset-1)	**Class**	A	B	C	D	E
	Instances	2	22	38	8	2
	Imbalanced Ratio	1	11	19	4	1
R. Asif et al. [16] (Dataset-2)	**Class**	A	B	C	D	E
	Instances	1	41	46	14	4
	Imbalanced Ratio	1	41	46	14	4
D Kabakchieva [17]	**Class**	Excellent	Very Good	Good	Average	Bad
	Instances	539	4336	4543	347	564
	Imbalanced Ratio	1.55	12.5	13.10	1	1.60

Unbalanced classes are a common issue in the classification of machine learning, where the number of findings is disproportionated in-class. Most algorithms for master learning work best if the sample numbers are approximately equal in each class [17]. Most algorithms have been developed to increase precision and decrease errors. Typically, the data imbalance represents an uneven class representation in a dataset. The fact that some classes have a slightly greater number of instances in the training set than certain classes is a typical issue in actual life implementations. Such a difference is called a class imbalance. Methods of addressing imbalances are well known for classical models of machine learning. Sampling methods are the most straightforward and common approach. Those methods work on the data itself (instead of the model) to increase its balance. The oversampling [18], is widely used and proven to be robust.

III. METHODOLOGY

The experiments have been performed in the Waikato Environment for Knowledge Analysis (WEKA) [19]. WEKA acknowledged as a landmark system in machine learning, and data mining has to turn into a widely used tool for data mining research [20]. Classifiers training is performed using 10-fold cross-validation [21]. To select classifiers, we first categorized and then choose one from each of the categories, probably, the one found frequently in literature. The meticulous findings are elaborated through the flow chart given in Figure 1.

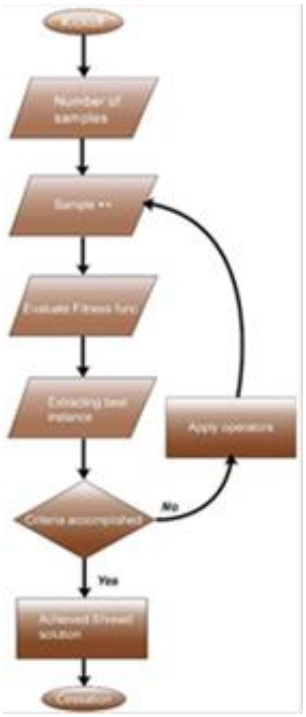

Figure 1. Information flow chart

Initially, from the data corpus, the samples are being collected based on the problem stated. All samples are applied in accordance with the mechanism described in the section of Classification with Imbalanced Dataset. Each sample will get incremented according to the required capacity. To balance the accuracy and manage the generated attributes that lead to new samples, the evaluation of the fitness function is carried out. The fitness feature is also calculated based on how many generations are made to prevent the overfitted classification model. When it accomplished the criteria, the final instances will be achieved. Otherwise, it will avail the operators like selection, crossover, and mutation and will again try to

maintain the balance condition by getting substantial increments, and so on [22].

Algorithm 1 as shown Figure 2 is extended to various datasets along with the sample previously collected, and rankings are therefore taken as the final accomplishment in keeping with the statement made in section 1.

Input: Sample size(Pop_Size), Selection method, Crossover_Rate (P_{cross}), Mutation_Rate (P_{mutate}), Max number of generation (Max_Gen), termination criteria, Fitness Function f(x)
Initialization: Generate Initial Random Sample
New_Pop, Pop = 0;
1: **for** $k = 1$ to *Pop_Size* **do**
2: Pop._add(_get_Random_Individual())
3: **end for**
// Evaluating the Fitness for all individual from Pop
Total_Fitness = 0;
4: **for** $i = 1$ to *Pop_Size* **do**
5: $F_{fitness}$ = pop[i].Evaluate() ——————————— (3)
6: Total_Fitness + = Ffitness
7: **end for**
// Perform elitism
8: **for** $i = 1$ to *_Elitism* **do**
9: New_Pop[count] = pop._getFittest()
10: count++ ;
11: **end for**
12: **while** *termination condition is not met* **do**
count = 0;
13: **while** *count < Pop_Size* **do**
//Apply Tournament Selection Method
14: Individual_1 = Pop.TournamentSelection()
15: Individual_2 = Pop.TournamentSelection()
16: **if** $P_{cross} \leq 0.9$ **then**
17: Indiv_1, Indiv_2 = Crossover(Individual_1, Individual2)
18: **end if**
19: **if** $P_{mutate} \leq 0.1$ **then**
20: Indiv_1.Mutate()
21: Indiv_2.Mutate()
22: **end if**
// Add to New_Sample
23: New_Pop.add(Indiv_1)
24: New_Pop.add(Indiv_2)
25: **end while**
Pop._Set_Sample(New_Pop)
// Evaluating the Fitness for all individual from Pop
Total_Fitness = 0;
26: **for** $i = 1$ to *Pop_Size* **do**
27: $F_{fitness}$ = pop[i].Evaluate() ——————————— (3)
28: Total_Fitness + = Ffitness
29: **end for**
30: **end while**
31: **return** *Pop*

Figure 2. Imbalance dataset selction mechanism

A. Memory Based Classifiers

In Memory Based Classifiers, the classification is based directly on the training examples. It stores the training set in the memory and then compares each instance with the instances it has seen in the training process. k-Nearest Neighbors (k-NN) [23], is an example of memory-based classifiers. It plots each instance as a point in multi-dimensional space and classifies it based on the class of their nearest neighbors.

B. Artificial Neural Network

This computational model is inspired by the structural and functional characteristics of the biological nervous system. Multilayer Perceptron (MLP) [24], is a class of Artificial Neural Networks.

C. Bayesian Statistics

Bayesian inference is a method of statistical inference [25]. It is based on using some evidence or observations in calculating the probability that a hypothesis may be correct, or besides update its previously-calculated probability [26].

D. Support Vector Machines (SVMs)

Support Vector Machines is a set of interrelated supervised learning methods that examine data and identify the patterns. Generally, Naïve Bayes and SVM algorithms are considered better choices for text classification [27].

E. Decision Tree

The decision tree [28], is a recursive technique that builds a tree. It starts with a root node, probably the essential attribute, branching through intermediate nodes and come to an end at the end node.

F. Performance Metrics

The Confusion Matrix, Precision, Recall, and F-Measure have been used to record the overall performance. Table 2 provides a standard visualization of a model having 2 class labels.

Table 2. Representation of standard confusion matrix

	Positive	Negative
Positive	True Positive (TP)	False Negative (FN)
Negative	False Positive (FP)	True Negative (TN)

Considering the "High" class as a (Positive) and "Low" class being a (Negative) term. Thereby rest of the terms are explained as:

- True Positive: Predicted as "High," and in fact, it is also "High."
- True Negative: Predicted as "Low," and in fact, it is also "Low."
- False Positive: Predicted as "High," but it is "Low."
- False Negative: Predicted as "Low," but it is "High."

- Recall

It is also called Sensitivity or True Positive Rate [29]. It is a measure of all positive instances and the number of instances the model predicted correctly. It is the ratio of positive cases that are predicted accurately and the actual number of positive examples that can be calculated, as shown in (1).

$$\text{Recall} = \text{TP}/(\text{TP} + \text{FN}) \quad (1)$$

- Precision

It shows all the positive instances that model has predicted correctly [30], how many are positively expressed by (2).

$$\text{Precision} = \text{TP}/(\text{TP} + \text{FP}) \quad (2)$$

- F-Measure

The values of recall and precision indicate the quality of the prediction model. However, it is sometimes not easy to make a decision based on precision and recall values. F-Measure takes both precision and recall in the account and calculates its weighted average [31]. It is calculated as given in (3).

$$\text{F} - \text{Measure} = 2\text{x}\left(\frac{\text{Precision x Recall}}{\text{Precision+Recall}}\right) \quad (3)$$

- Accuracy

It is the ratio of the sum of TP and TN and the total number of instances expressed in (4).

$$\text{Accuracy} = (\text{TP} + \text{TN})/n \quad (4)$$

Where n is the total number of instances in the dataset.

G. Dataset

The dataset contains 151 instances, which are the total number of students enrolled in a core course "CMP427" during the three semesters taught in the IT department at Al-Buraimi University College, Buraimi, Sultanate of Oman. Final Grade is the prediction feature with "Low" and "High" classes. Usually, the students frequently obtaining grades below 65% are considered at the risk of losing academic benefits.

- Undersampling

This method is applied to most classes. Undersampling reduces the instances in the majority class to make them approximately equal to the cases in the minority class. Spread Subsampling is one of the undersampling algorithms which we use in this research. Spread Subsampling creates a random subsample of the imbalanced dataset. It adjusts the class distribution by randomly eliminating instances from the majority class [32]. To compute the distribution, Spread Subsampling takes Spread-Distribution value (a parameter) from the user who specifies the maximum ratio between the classes.

- Oversampling

Synthetic Minority Over-sampling Technique (SMOTE) [33], oversamples the minority class with random under-sampling of the majority class. This algorithm rebalances the original training set by conducting an oversampling approach. A SMOTE forms new instances for minority class by interpolating among several minority class instances that recline together. The k-nearest neighbors of minority class instances are computed, and afterward, particular neighbors are selected. New synthetic data samples are generated from these neighbors. SMOTE does not change the number of instances in the majority class. SMOTE has a parameter (percentage), which specifies the desired increase in the minority class.

H. Understanding oversampling and undersampling at the algorithm level

The Synthetic Minority Oversampling (SMOTE) is an oversampling technique that synthetically produces instances by arbitrarily selecting minority class instances and using interpolation methods to create instances between the selected point and its neighboring instances. Through this process, any instance of a minority class is considered, and new instances of a minority class are created along the line segment joining its nearest neighbours. The number of synthetic instances is generated based on the requisite percentage of oversampling. The Algorithm steps are as follows:

- Load data collection, and classify the division of minority and majority;
- Calculate the number of instances to be generated using the oversampling percentage;
- Identify a minority class random case, and locate its closest neighbours;
- Choose one of the nearest neighbors and find the difference between random instance and neighbor selected;
- Multiply the difference by a number generated at random between 0 and 1;
- Add that difference to the chosen instance at random;
- Repeat the cycle from three to six until it produces the number of instances according to the percentage given.

Further, Random Undersampling (RUS) is a simple undersampling strategy that excludes instances at random from the main class of Align the dataset before classification methodology is applied. The main challenge of this strategy is that it can eliminate the relevant details in the dominant class

that might not be appropriate in certain situations. The operating steps are as follows.

- Launch the dataset and classify the minority and majority classes;
- Calculate the number of instances to be removed based on the percentage of undersampling;
- Identify a random instance in the majority class and delete it from the majority class;
- Repeat step three until the number of instances eliminated is equal to the specified percentage.

IV. RESULT AND DISCUSSION

Most of the classifiers tend to maximize accuracy, despite higher accuracy; a classifier may produce unsatisfactory results, given that the training dataset is imbalanced. In an ideal dataset, the number of instances in the classes is equal. The Imbalance Ratio (IR) expresses how imbalanced a dataset is and is defined as the ratio of the sizes of majority and the minority class. The dataset having IR=1 is balanced, and thus the dataset with higher IR is more imbalanced. Imbalanced classes bias the classifiers, which tend to classify all instances into the majority class. Data Balancing refers to decreasing the value of IR and bring it close to one. The other literature shows that tuning class distribution can improve classifier performance. Though there is no unified rule for class balancing, it can still be inferred that classification with sampling techniques yielded optimal results than going without them. Over time, several algorithms have been developed to deal with the class imbalance problem. The data-level algorithms make use of sampling techniques to adjust the Imbalance Ratio. They are grouped as an oversampling and under-sampling algorithm. Oversampling methods increase the number of instances in the minority class to balance the classes; in contrast, under sampling remove instances from the majority class to adjust the class distribution. Figure 3 depicts the idea of both under sampling and oversampling algorithms.

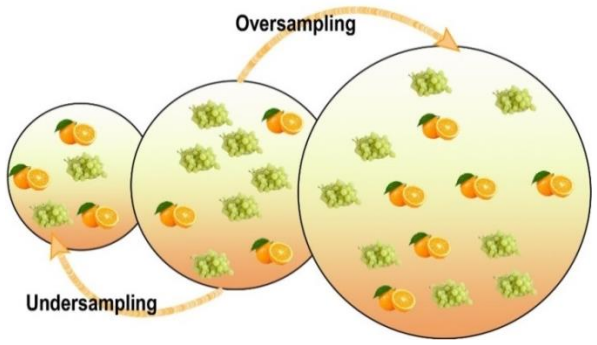

Figure 3. Depiction of oversampling and under sampling

The center dataset is imbalanced with graps in majority class; the left-side illustrates dataset after under sampling where instances are removed from graps class while orange instances are added to the right side when oversampling is performed. Imbalanced dataset from an educational environment having an Imbalance Ratio (IR=1:3.19). Around two-thirds of the instances are from the majority (High) class comparing to the low number of instances in the minority (Low) class.

A. Classification with Imbalanced Dataset

We perform classification with an imbalance dataset to compare the accuracy of the classifiers with other performance evaluation measures. Table 3 shows the result obtained from the classifiers. It outlines the accuracy of each classifier. Further, it provides Precision, Recall, and F-Measure for minority (Low), majority (High) classes, and also their average. The last column contains the confusion matrix for each classifier.

Table 3. Results from classification with imbalanced dataset

Classifier	Accuracy	Classes	Precision	Recall	F-Measure	Confusion Matrix
Naïve Bayes	84.77%	Low	0.659	0.750	0.701	a b <-- classified as 27 9 | a = Low 14 101 | b = High
		High	0.918	0.878	0.898	
		Average	0.856	0.848	0.851	
Multilayer Perceptron	80.79%	Low	0.600	0.583	0.592	a b <-- classified as 21 15 | a = Low 14 101 | b = High
		High	0.871	0.878	0.874	
		Average	0.806	0.808	0.807	
Support Machine Vector (SMO)	89.40%	Low	0.833	0.694	0.758	a b <-- classified as 25 11 | a = Low 5 110 | b = High
		High	0.909	0.957	0.932	
		Average	0.891	0.894	0.891	
IBk	78.81%	Low	0.559	0.528	0.543	a b <-- classified as 19 17 | a = Low 15 100 | b = High
		High	0.855	0.870	0.862	
		Average	0.784	0.788	0.786	
Random Forest	86.09%	Low	0.727	0.667	0.696	a b <-- classified as 24 12 | a = Low 9 106 | b = High
		High	0.898	0.922	0.910	
		Average	0.858	0.861	0.859	

The results show that most of the classifiers have produced more than 80% accuracy. The confusion matrix gives an idea of how many instances of each class are misclassified by each classifier. Figure 4 shows a chart that compares the accuracy (data labels at the top of the bar) of each classifier and the F-Measure (in percent) for both minority (data labels at the center of the bar) and majority (data labels at the bottom of the bar) classes. Despite achieving higher accuracies and F-Measures for the majority class, the classifiers have made relatively lower F-Measure values for the minority class. For instance, Support Vector Machine (SMO) has exceptionally low F-Measure for minority (75.8%) class comparing to its high accuracy (89.4%).

Moreover, the difference between F-Measure of majority and minority class is high for all classifiers. It concludes the bias behavior of classifiers over an imbalanced dataset. The classifiers achieved reasonably high accuracy but failed to classify the minority class instances correctly.

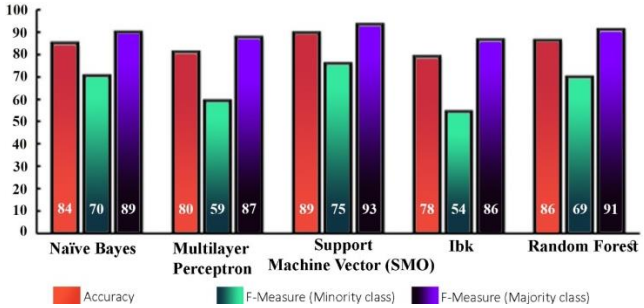

Figure 4. Accuracy comparison, the F-Measures of classifiers for minority and majority class over the imbalanced dataset

B. Undersampling dataset classification

We apply the Spread Subsampling algorithms, an under-sampling algorithm for balancing the imbalanced dataset. Figures 5 and 6 illustrate the impact of Spread Subsampling produced datasets. Similarly, Table 4, thereby shows the performance measures of classifiers when Spread Subsampling is implemented. Both Support Vector Machine (SMO) and Multilayer Perceptron achieve the highest accuracy. Multilayer Perceptron produces slightly higher F-Measure and Recall values for the minority class. The reason behind this can be seen in the confusion matrix, which shows that Multilayer Perceptron misclassifies only four instances of minority class comparing to Support Vector Machine (SMO) with five.

To compare classification with imbalanced datasets and under-sampled datasets, A chart is illustrated in Figure 7, which presages the decrease in the accuracy of classifiers (except Multilayer Perceptron) after under-sampling. This may indicate that the classifiers have reduced partiality and are correctly classifying instances.

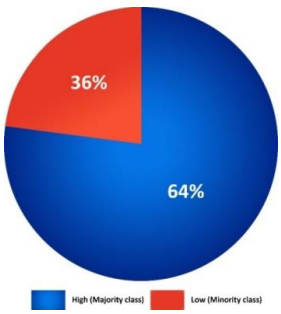

Figure 5. An imbalanced dataset

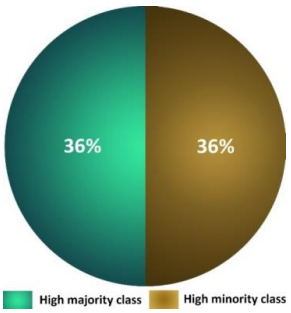

Figure 6. Balanced dataset

Table 4. Results from classification after undersampling

Classifier	Accuracy	Classes	Precision	Recall	F-Measure	Confusion Matrix
Naïve Bayes	81.94%	Low	0.795	0.861	0.827	a b <-- classified as 31 5 | a = Low 8 28 | b = High
		High	0.848	0.778	0.812	
		Average	0.822	0.819	0.819	
Multilayer Perceptron	86.11%	Low	0.842	0.889	0.865	a b <-- classified as 32 4 | a = Low 6 30 | b = High
		High	0.882	0.833	0.857	
		Average	0.862	0.861	0.861	
Support Machine Vector (SMO)	86.11%	Low	0.861	0.861	0.861	a b <-- classified as 31 5 | a = Low 5 31 | b = High
		High	0.861	0.861	0.861	
		Average	0.861	0.861	0.861	
IBk	76.39%	Low	0.757	0.778	0.767	a b <-- classified as 28 8 | a = Low 9 27 | b = High
		High	0.771	0.750	0.761	
		Average	0.764	0.764	0.764	
Random Forest	83.33%	Low	0.833	0.833	0.833	a b <-- classified as 30 6 | a = Low 6 30 | b = High
		High	0.833	0.833	0.833	
		Average	0.833	0.833	0.833	

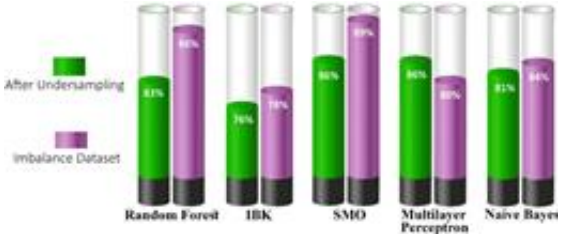

Figure 7. Performance comparison before and after undersampling

C. Oversampling dataset classification

The SMOTE has been utilized to balance datasets through oversampling. Keeping 200 as the percentage value, SMOTE approached 108 instances of the minority class. Figure 8 shows the class distribution after oversampling.

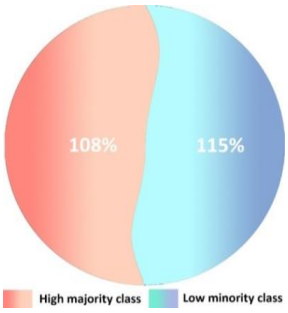

Figure 8. After oversampling **c**lass distribution

SMOTE appends the newly created instances at the end of the dataset file. Since we are using k-fold cross-validation, this possibly will give rise to data over-fitting. To avoid over-fitting, we randomized the instances in our dataset to have a dataset with randomly distributed instances. Table 5 provides results for classification after oversampling. The application of SMOTE has further enhanced the performance of the classifier. Multilayer Perceptron has achieved the highest accuracy.

Table 5. Classification results after oversampling

Classifier	Accuracy	Classes	Precision	Recall	F-Measure	Confusion Matrix
Naive Bayes	87.89%	Low	0.852	0.907	0.879	a b <-- classified as 98 10 | a = Low 17 98 | b = High
		High	0.907	0.852	0.879	
		Average	0.881	0.879	0.879	
Multilayer Perceptron	91.03%	Low	0.873	0.954	0.912	a b <-- classified as 103 5 | a = Low 15 100 | b = High
		High	0.952	0.870	0.909	
		Average	0.914	0.910	0.910	
Support Machine Vector (SMO)	88.79%	Low	0.849	0.935	0.890	a b <-- classified as 101 7 | a = Low 18 97 | b = High
		High	0.933	0.843	0.886	
		Average	0.892	0.888	0.888	
IBk	83.86%	Low	0.805	0.880	0.841	a b <-- classified as 95 13 | a = Low 23 92 | b = High
		High	0.876	0.800	0.836	
		Average	0.842	0.839	0.838	
Random Forest	90.13%	Low	0.898	0.898	0.898	a b <-- classified as 97 11 | a = Low 11 104 | b = High
		High	0.904	0.904	0.904	
		Average	0.901	0.901	0.901	

The chart in Figure 9, compares classifiers' performance using average F-Measure after oversampling. This chart confirms that the average F-measure for classifiers has increased with oversampling for both datasets. Figure 10 highlights an increase in the precision (in percent) of minority class with oversampling. This chart illustrates that oversampling has increased the precision of the minority class. The highest increase is achieved by Multilayer Perceptron, and the lowest is produced by SMO.

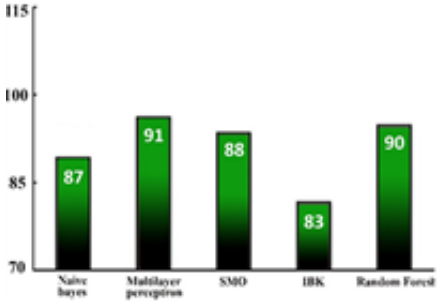

Figure 9. Performance comparison using average F-Measure

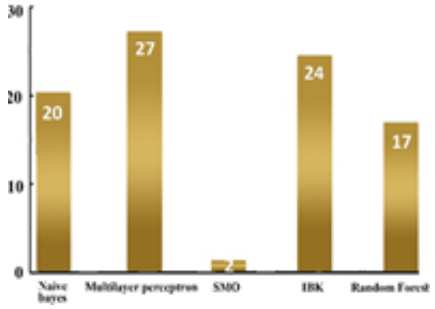

Figure 10. Precision increasing with oversampling

V. OUTCOME IN A GLANCE

The outcome shows that it's not only the accuracy of a classifier that decides whether it is predicting well. Other performance measures, such as F-Measure, Precision, and Recall value for minority class, should be observed as well. These findings support the argument that the classifiers with an imbalanced dataset tend to misclassify most of the instances as majority class. It is noted that both under sampling and oversampling algorithms are effective in decreasing the difference between the F-Measures of majority and minority classes. In both cases, the classifiers achieved reasonable accuracies and F-Measure values. However, it also showed that between the two sampling algorithms oversampling (SMOTE) has performed better than under sampling. The oversampling approach shows superiority over under-sampling Smote.

VI. CONCLUSION

Comparative performance of classifiers with imbalanced datasets to the dataset, which is balanced with oversampling and under sampling algorithms, has been performed. The dataset is taken from an educational context. The classifiers are categorized in different categories, and one classifier is selected from each category. We conclude that classification with an imbalanced dataset may produce higher accuracy, but low F-Measure values for the minority class. This shows that the classifiers misclassify the minority class instances. We applied under sampling (Spread Subsampling) and oversampling (SMOTE) upon our dataset. It shows that both Spread Subsampling and SMOTE increases the F-Measure values for the minority class. However, it indicates that SMOTE performs better than Spread Subsampling in achieving higher F-Measure value and accuracy.

VII. REFERENCES

[1] M. A. U. H. Tahir, S. Asghar, A. Manzoor, and M. A. Noor, "A Classification Model For Class Imbalance Dataset Using Genetic Programming," IEEE Access, vol. 7, pp. 71013–71037, 2019, doi: 10.1109/ACCESS.2019.2915611.

[2] S. Ashraf, M. Gao, Z. Chen, S. Kamran, and Z. Raza, "Efficient Node Monitoring Mechanism in WSN using Contikimac Protocol," Int. J. Adv. Comput. Sci. Appl., vol. 8, no. 11, 2017, doi: 10.14569/IJACSA.2017.081152.

[3] I. Khan, A. Al Sadiri, A. R. Ahmad, and N. Jabeur, "Tracking Student Performance in Introductory Programming by Means of Machine Learning," in 2019 4th MEC International Conference on Big Data and Smart City (ICBDSC), Jan. 2019, pp. 1–6, doi: 10.1109/ICBDSC.2019.8645608.

[4] S. Ashraf, A. Raza, Z. Aslam, H. Naeem, and T. Ahmed, "Underwater Resurrection Routing Synergy using Astucious Energy Pods," J. Robot. Control JRC, vol. 1, no. 5, 2020, doi: 10.18196/jrc.1535.

[5] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," ACM SIGKDD Explor. Newsl., vol. 11, no. 1, pp. 10–18, Nov. 2009, doi:

10.1145/1656274.1656278.
[6] 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, doi: 10.1016/j.patcog.2006.01.009.
[7] "Illustration of a Tomek link imbalanced learning." https://imbalanced-learn.readthedocs.io/en/stable/auto_examples/under-sampling/plot_illustration_tomek_links.html (accessed Jun. 16, 2020).
[8] S. Ashraf et al., "Underwater Routing Protocols Analysis of Intrepid Link Selection Mechanism, Challenges, and Strategies," Int. J. Sci. Res. Comput. Sci. Eng., vol. 8, no. 2, pp. 1–9, Apr. 2020, doi: 10.26438/ijsrcse/v8i2.19.
[9] D. Kabakchieva, "Predicting Student Performance by Using Data Mining Methods for Classification," Cybern. Inf. Technol., vol. 13, no. 1, pp. 61–72, Mar. 2013, doi: 10.2478/cait-2013-0006.
[10] O. Scheuer and B. M. McLaren, "Educational Data Mining," in Encyclopedia of the Sciences of Learning, N. M. Seel, Ed. Boston, MA: Springer US, 2012, pp. 1075–1079.
[11] S. Ashraf, Z. A. Arfeen, M. A. Khan, and T. Ahmed, "SLM-OJ: Surrogate Learning Mechanism during Outbreak Juncture," Int. J. Mod. Trends Sci. Technol., vol. 6, no. 5, pp. 162–167, May 2020, doi: 10.46501/IJMTST060525.
[12] Pristyanto and A. Dahlan, "Hybrid Resampling for Imbalanced Class Handling on Web Phishing Classification Dataset," in 2019 4th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE), Yogyakarta, Indonesia, Nov. 2019, pp. 401–406, doi: 10.1109/ICITISEE48480.2019.9003803.
[13] S. Sasikala, S. Appavu alias Balamurugan, and S. Geetha, "Multi Filtration Feature Selection (MFFS) to improve discriminatory ability in clinical data set," Appl. Comput. Inform., vol. 12, no. 2, pp. 117–127, Jul. 2016, doi: 10.1016/j.aci.2014.03.002.
[14] S. Fatima and S. Mahgoub, "Predicting Student's Performance in Education using Data Mining Techniques," Int. J. Comput. Appl., vol. 177, no. 19, pp. 14–20, Nov. 2019, doi: 10.5120/ijca2019919607.
[15] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," Neural Netw., vol. 106, pp. 249–259, Oct. 2018, doi: 10.1016/j.neunet.2018.07.011.
[16] W. Xie, G. Liang, Z. Dong, B. Tan, and B. Zhang, "An Improved Oversampling Algorithm Based on the Samples' Selection Strategy for Classifying Imbalanced Data," Math. Probl. Eng., vol. 2019, pp. 1–13, May 2019, doi: 10.1155/2019/3526539.
[17] S. Ashraf, M. Gao, Z. Mingchen, T. Ahmed, A. Raza, and H. Naeem, "USPF: Underwater Shrewd Packet Flooding Mechanism through Surrogate Holding Time," Wirel. Commun. Mob. Comput., vol. 2020, pp. 1–12, Mar. 2020, doi: 10.1155/2020/9625974.
[18] J. Hernandez, J. A. Carrasco-Ochoa, and J. F. Martínez-Trinidad, "An Empirical Study of Oversampling and Undersampling for Instance Selection Methods on Imbalance Datasets," in Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, Berlin, Heidelberg, 2013, pp. 262–269.
[19] Y. Liu, Y. Wang, X. Ren, H. Zhou, and X. Diao, "A Classification Method Based on Feature Selection for Imbalanced Data," IEEE Access, vol. 7, pp. 81794–81807, 2019, doi: 10.1109/ACCESS.2019.2923846.
[20] [S. Ashraf, T. Ahmed, S. Saleem, and Z. Aslam, "Diverging Mysterious in Green Supply Chain Management," Orient. J. Comput. Sci. Technol., vol. 13, no. 1, pp. 22–28, May 2020, doi: 10.13005/ojcst13.01.02.
[21] A. Arshad, S. Riaz, and L. Jiao, "Semi-Supervised Deep Fuzzy C-Mean Clustering for Imbalanced Multi-Class Classification," IEEE Access, vol. 7, pp. 28100–28112, 2019, doi: 10.1109/ACCESS.2019.2901860.
[22] S. Ashraf, T. Ahmed, A. Raza, and H. Naeem, "Design of Shrewd Underwater Routing Synergy Using Porous Energy Shells," Smart Cities, vol. 3, no. 1, pp. 74–92, Feb. 2020, doi: 10.3390/smartcities3010005.
[23] H. Zhang, Z. Li, H. Shahriar, L. Tao, P. Bhattacharya, and Y. Qian, "Improving Prediction Accuracy for Logistic Regression on Imbalanced Datasets," in 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC), Milwaukee, WI, USA, Jul. 2019, pp. 918–919, doi: 10.1109/COMPSAC.2019.00140.
[24] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," ACM SIGKDD Explore. Newsl., vol. 6, no. 1, pp. 20–29, Jun. 2004, doi: 10.1145/1007730.1007735.
[25] S. Ashraf, S. Saleem, A. H. Chohan, Z. Aslam, and A. Raza, "Challenging strategic trends in green supply chain management," Int. J. Res. Eng. Appl. Sci. JREAS, vol. 5, no. 2, pp. 71–74, 2020.
[26] "Bayesian Statistics," Analytics Vidhya, Jun. 20, 2016. https://www.analyticsvidhya.com/blog/2016/06/bayesian-statistics-beginners-simple-english/ (accessed Jun. 16, 2020).
[27] S. Ashraf, S. Saleem, and T. Ahmed, "Sagacious Communication Link Selection Mechanism for Underwater Wireless Sensors Network," Int. J. Wirel. Microw. Technol., vol. 10, no. 2, pp. 12–25.
[28] J. F. Magee, "Decision Trees for Decision Making," Harvard Business Review, no. July 1964, Jul. 01, 1964.
[29] S. Ashraf, S. Saleem, T. Ahmed, and M. A. Khan, "Multi-biometric Sustainable approach for human Appellative," Trends Comput Sci Inf Technol, vol. 5, no. 1, pp. 001–007.
[30] "Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures." https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/ (accessed Jun. 16, 2020).
[31] S. Ashraf, A. Yahya, and M. A. Khan, "Culminate coverage for sensor network through Bodacious-instance Mechanism," -Manag. J. Wirel. Commun. Netw., vol. 8, no. 3, pp. 1–7.
[32] E. R. Q. Fernandes, A. C. P. L. F. de Carvalho, and X. Yao, "Ensemble of Classifiers Based on Multiobjective Genetic Sampling for Imbalanced Data," IEEE Trans. Knowl. Data Eng., vol. 32, no. 6, pp. 1104–1115, Jun. 2020, doi: 10.1109/TKDE.2019.2898861.
[33] B. S. Raghuwanshi and S. Shukla, "SMOTE based class-specific extreme learning machine for imbalanced learning," Knowl.-Based Syst., vol. 187, p. 104814, Jan. 2020, doi: 10.1016/j.knosys.2019.06.022.