IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 13, 2020, accepted November 8, 2020, date of publication November 17, 2020,
date of current version December 15, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3038658

Empirical Comparison of Approaches for Mitigating Effects of Class Imbalances in Water Quality Anomaly Detection

EUSTACE M. DOGO[1], NNAMDI I. NWULU[1], BHEKISIPHO TWALA[2], (Senior Member, IEEE), AND CLINTON OHIS AIGBAVBOA[3]

[1]Department of Electrical and Electronic Engineering Science, University of Johannesburg, Johannesburg 2006, South Africa
[2]Faculty of Engineering and the Built Environment, Durban University of Technology, Durban 4000, South Africa
[3]Sustainable Human Settlement and Construction Research Centre, Faculty of Engineering and the Built Environment, University of Johannesburg, Johannesburg 2006, South Africa

Corresponding author: Eustace M. Dogo (eustaced@uj.ac.za)

This work was supported by the University of Johannesburg, South Africa.

ABSTRACT Imbalanced class distribution and missing data are two common problems and occurrences in water quality anomaly detection domain. Learning algorithms in an imbalanced dataset can yield an overrated classification accuracy driven by a bias towards the majority class at the expense of the minority class. On the other hand, missing values in data can induce complexity in the learning classifiers during data analysis. These two problems pose substantial challenges to the performance of learning algorithms in real-life water quality anomaly detection problems. Hence, the need for them to be carefully considered and addressed to achieve better performance. In this paper, the performance of a range of several combinations of techniques to deal with imbalanced classes in the context of binary-imbalanced water quality anomaly detection problem and the presence of missing values is extensively compare. The methods considered include seven missing data and eight resampling methods, on ten different learning state-of-the-art classifiers taking into account diversity in their learning philosophies. The different classifiers are evaluated using stratified 5-fold cross-validation, based on three performance evaluation metrics namely accuracy, ROC-AUC and F1-measure. Further experiments are carried out on nineteen variants of homogeneous and heterogeneous ensemble techniques embedded with resampling and missing value strategies during their training phase as well as an optimized deep neural network model. The experimental results show an improvement in the performance of the learning classifiers, especially when dealing with the class imbalance problem (on the one hand) and the incomplete data problem (on the other hand). Furthermore, the neural network model exhibit superior performance when dealing with both problems.

INDEX TERMS Class-imbalance, data preprocessing, imputation, machine learning, resampling, water quality.

I. INTRODUCTION

There is a consensus that easy access to water of good quality to the public leads to improved health and living conditions, and has a direct impact on the economy and national security of countries. Furthermore, due to the massive amount of data currently generated by water utilities and the impact of the water industry on the lives of people [1]. There is a need to implement better ways of water quality monitoring and prediction based on new and advanced technologies such as new and enhanced machine learning and data mining techniques [2]. Imbalanced class distribution (ICD) and missing values (MV) in data are two common problems and occurrences in data analysis that are synonymous with data quality issues [3]–[5]. MV and ICD continue to be prevalent in numerous real-world problems and across many application areas [6], [7], including water quality anomaly detection domain. Consequently, these occurrences have continued to generate lots of attention from researchers because the majority of conventional predictive machine learning algorithms are not developed to handle these challenge in data, because they assume completeness of data and a balanced

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

class distribution [8], [9]. As a result, predictive or classification algorithms perform sub-optimally on these kinds of datasets if not properly handled, resulting in bias, inaccurate and low-quality predictive performance of the classifiers [7], [8], [10]. Similarly, previous investigations of missing values and imbalanced class distribution challenges in water quality anomaly detection have taken place mostly in isolation, even though their harmful effects on the classifiers' performance is well acknowledged in many research works. Although some authors have experimented using a combination of both methods, their effects and interactions on learning algorithms were not fully showcased. This study aims to demonstrate that considering the effects of both missing values and imbalanced class distribution in water quality anomaly detection offers a means of better understanding the problems, and thereby offering ways of mitigating their effects on the performance of learning algorithms. This study also demonstrates experimentally and verify our hypotheses that these two problems harm the performance measures of learning classifiers, and hence a reason to consider them in tandem because of their prevalence in the examined domain. Additionally, the imbalanced class problem cannot be possibly considered solved without considering the challenge pose by missing values in data since it also harms the performance of learning classifiers and often occur together with class imbalance distribution.

Class imbalance is a term that refers to a dataset that has an uneven distribution of classes, whereby one or more of the classes have a larger number of instances than the other does. In a binary-class scenario, for example, the class with the most frequent instances is referred to as majority class, while the class with the rarest instances is the minority class. Anomalies in water quality are rare events of interest in real-life, but predicting these rare events from an imbalance learning perspective using traditional machine learning approaches poses significant challenges to researchers [11]. Reasons for such challenges would include the inability of the traditional classifiers to cope with imbalanced scenarios. As a result of problems that include treatment of rare events as noise, overlapping of minority and majority classes, biases induced by performance metrics such an accuracy towards the majority class and disjuncts in imbalanced data that entails small sample size with a high feature dimensionality [12].

Missing data are prevalent in almost all of the research domains that relays on sensors for data generation such as in monitoring the quality of water in an urban water distribution network. Missing data is a term that refers to the absence of values or observations, which are usually anticipated to be present in a dataset [4]. Missing data in water quality anomaly detection dataset can arise for several reasons, such as faulty sensor readings and measurement errors that could be as a result of low signal-to-noise ratio during digital signal processing, mistakes and mishandling of data during generation or reporting by personnel, and sometimes the outright deletion of data information [13]. Approaches to handling missing data are well documented in the literature, using statistical or machine learning methodologies [14]. The process of replacing or substituting missing values are collectively termed as missing data imputation [13]. Missing values if not adequately addressed induces an element of complexity into data analysis, and not only affect the performance of machine learning (ML) algorithms, but also impacts on the value that could be derived in terms of accurately detecting an anomaly in the water distribution system [13].

The importance of data reliability and completeness in ensuring data quality cannot be overemphasised, particularly due to their relevance in enhancing the predictive performance of learning algorithms, and by extension the value of the information that can be derived from the data [5]. It is for these reasons that motivate researchers on the need to address them, to produce a more reliable outcome or conclusions that can be inferred from a dataset [7], [10]. There are numerous reported strategies or methods in scholarly works that have been developed to deal with incomplete data and class-imbalance problems [6], [7].

Several solutions have been proposed in the literature in dealing with water quality anomaly detection. For example, a study using tree-based ensemble approaches is carried out in [15]. Several machine learning and deep learning approaches to dealing with anomaly detection in water quality based on time-series data are examined in [16]. Multi-objective machine learning for feature selection on support vector machine and ensemble generation on decision trees is proposed in [17] to solve online anomaly detection of drinking-water quality on time series data. Authors in [18] further proposed two imbalance boosting-based ensemble models namely SMOTEBoost and RUSBoost using oversampling and undersampling techniques to balance the training data respectively. The authors finally applied multi-objective pruning on the base models for the ensembles, to optimize the prediction and generalisation performance of their models. The authors in [19], proposed two models namely adaptive learning rate BP neural network and 2-step isolation, and random forest to predict water quality based on both physical and biological indicators in an urban water supply scenario. Most of these works focused on specific missing values and class-imbalance methods in dealing with the challenges in this domain. We also observe that majority of these works focused on evaluating only the training set, preprocessed with MV and ICD methods. However, applying MV and ICD methods on the training set is one case, while evaluating the classifiers on the imbalanced test set (unseen data) with or without MV is a different case altogether.

This work experimentally studies the mitigating effects of applying missing data and imbalanced class distribution methods in water quality anomaly classification problem based on two hypotheses that 1) missing data does harm performance measures; 2) class-imbalances would harm performance measures. In this paper, we conduct an exploratory study to compare the performance of select MV, data-level ICD and ensemble approaches previously proposed in the literature on different classifiers. Specifically,

a combination of seven MV methods (replacing missing values with zero, listwise deletion, mean, mode, missForest, expectation-maximization (EM), and multiple imputations by chained equations (MICE)), and eight resampling methods (ROS, SMOTE, ADASYN, RUS, Tomek links, RENN, SMOTE + Tomek links, SMOTE + ENN) on the performance of ten different classifiers (LR, k-NN, LDA, SVC, NB, DT, RF, AdaBoost, ANN and DNN). Furthermore, we empirically evaluate the performance of 19 static ensembles of heterogeneous and homogeneous approaches that include bagging, boosting, stacking and their variants embedded with resampling strategies during their training phase, and as well as an optimized DNN model. To the best of our knowledge, such a comprehensive experimental study of the combination of several techniques for anomaly detection in drinking-water quality classification problem is not common. Furthermore, the comparison of these models would provide useful insight and shed more light on their benefits and differences. It is worth noting that the purpose of this study is not to examine all existing methods but to focus on methods that are frequently presented in the literature for our dataset problem.

The three major contributions and objective of this study are given as follows:

1) To compare the effect of 7 missing data algorithms on 10 different machine learning models.
2) To compare the effect of combining of 7 missing data and 8 resampling methods on 10 different machine learning models;
3) To investigate the performance of different formulations of machine learning algorithms on imbalanced class distribution and missing data, that include different ensembles and deep neural network models.

The rest of the paper is organized as follows. Sections II presents a brief overview of classifiers, resampling and missing values methods. The experimental setup, dataset and the performance metrics are reported in section III. Section IV reports the experimental results, including statistical tests and time computational complexity of the models. A discussion that summarizes the experimental findings is reported in section V. Lastly, section VI concludes the paper.

II. OVERVIEW OF METHODS

A. CLASSIFIER

1) CLASSIFICATION PROBLEM DEFINITION

A classifier is a mathematical function that assigns class labels to training data instances [20]. Given a dataset with test features $X_{Test} \in /R^{n*k}$ are a matrix of n test examples and k features and vector $y_{Test} \in \{0, 1\}^n$ classifying False (0) or True (1) of an event. The aim is to estimate a function f such that $y = f(x)$ which minimises the misclassification as $\min \sum_{i=1}^{n} (y_i - f(x_i))^2$, where $f(x_i)$ is the prediction for the i^{th} test example and y_i is the i^{th} classification.

Depending on the assumptions of the learning model, classifiers can be categorized as either parametric (fixed number of parameters for data distribution) and nonparametric (data distribution with no fixed number of parameters) [21]. Ten supervised learning classifiers are considered in this study. They are selected based on their appropriateness for our dataset problem and their different learning philosophies (linear, density-based, instance-based, tree-based and neural network-based models), in order to consider a broad spectrum of families of learning algorithms [22]. This ensures a robust assessment of the effects of the missing data and the class-imbalanced methods on the selected classifiers evaluated in this study [3], [7], [8], [14], [23].

B. RESAMPLING METHOD

Resampling methods aim to transform a dataset distribution to account for the imbalanced or skewness nature of the class labels in classification tasks, in order to arrive at a fairer and acceptable decision boundary [6]. Numerous resampling techniques in literature are mostly k-NN or Euclidean distance inspired, and can be broadly categorized into four: 1) over-sampling the minority class, 2) under-sampling the majority class, 3) hybrid combination of under-sampling used in conjunction with over-sampling methods and 4) creating an ensemble with balanced dataset [6]. Another resampling categorization usually adopted is based on methods that consider and select the data examples to keep, methods that consider and select examples to delete and the hybrid of both. Since in this study we are concerned with binary classification problem, throughout this paper we shall refer to the majority class (True) or normal state as class 0 and the minority class (False) or abnormal state as class 1. In this study, the occurrence of the minority class 1 that is poorly represented in the data space in comparison to class 0 represents the class of interest.

Class imbalanced solutions are broadly categorized into four different approaches [10], [11], [24]:

1) DATA-LEVEL

This approach addresses the class-imbalanced problem by resampling class distribution during preprocessing. Techniques that perform these class modifications are collectively known as resampling algorithms. The resampling algorithms handle class-imbalance problems, by either over-sampling the minority class, under-sampling the majority class or a hybrid approach of combining over-sampling and under-sampling.

2) ALGORITHM-LEVEL

This approach involves learning algorithms adapted to handle imbalanced class distribution, by modifying the learning algorithms to handle such a problem. An example of an algorithm level approach is where an ensemble classification method incorporates an internally resampling technique before creating the ensembles.

3) COST-SENSITIVE

The algorithms in this approach take into account the cost associated with the different class instances (minority or

majority classes) by assigning a different cost, in the process, the learning algorithm is modified to take into account the assigned costs. For example, a high misclassification cost is assigned to the minority class during the learning process to underscore its importance as the class of interest, weakening the majority class in the process. The cost-sensitive approach could either use a direct method, whereby the cost is assigned directly on the class instances or through a meta-learning approach that employs during training a data-level technique for prepossessing or employing some postprocessing steps.

4) MULTIPLE CLASSIFIERS ENSEMBLE (MCE)

This approach involves combining an ensemble learning algorithm with either one of data-level, cost-sensitive or algorithm-level approach during preprocessing.

C. MISSING DATA METHOD

Missing data method (MDM) is a form of data cleaning process that usually forms part of data preprocessing [5]. MDM could be broadly categorised into two strategies for handling incomplete data [5], [25]: 1) Missing data toleration strategy that ignores, delete or remove missing values in either the training or test dataset. 2) Missing data imputation which entails filling missing values in a dataset with some suitable and estimated values, rather than leaving them empty. Gaining an understanding of the pattern and mechanism for missing values is a critical step that will inform the type of strategy to use in any given scenario [4], in addition to knowing the percentage of missing values in data and the sample size [23]. The mechanism for missing values are broadly categorized into three: missing completely at random (MCAR), missing at random (MAR) and missing not at random (MNAR) [4]. The assumption on the nature or mechanism of missing values in data could be derived by understanding the data collection process, as well as through statistical investigation and testing [23]. Different missing value methods induce biases, and uncertainties, particularly if the methods are based on certain assumptions concerning the missing value mechanisms. It is also worth noting that mean and mode imputations are reported in the literature to induce more uncertainties during missing values estimations as compared to the MICE method. This is for example because MICE take into account all the available information from other instances in the data and then averages their results to provide better estimates of the unknown true missing value. Various strategies for handling missing values in data exist in the literature, they include statistical, machine learning, model-based using maximum likelihood with EM, and ensemble approaches [7]. Missing data strategies are broadly categorized into four: (1) Case deletion - filling with a value, or ignoring data with missing values, or deleting or dropping missing values, (2) imputation strategies (mean, median, multiple imputation & machine learning such as k-NN, (3) model-based imputation strategies (maximum-likelihood with EM algorithm) and (4) machine learning-based strategies (ensemble approach with RF) [7].

With the increased availability of computational resources, more complex and advanced missing data techniques have become available. Recent development as candidate solutions of missing data recovery task is using computational and artificial intelligence approaches to address identified disadvantages such as low prediction performance in the well-known missing data methods. In [26], the authors proposed a computational intelligence technique, the non-iterative neural-like structures of the Successive Geometric Transformation Model (SGTM) to handle missing data. The authors reported an increased estimation accuracy in comparison with the arithmetic mean algorithm. The authors in [27], proposed a solution to missing values using the General Regression Neural Network (GRNN). The method is reported to show improved performance accuracy when compared to previous methods. To further improve the performance accuracy of missing values specifically in data collected through IoT devices, the authors in [28], proposed an ensemble method (GRNN-SGTM) by combining GRNN network and SGTM neural-like structure. The performance of the ensemble method was shown to be more effective in comparison with single standalone GRNN and SGTM methods. The combination of neural network and Evolutionary computing has also been well studied in literature as an effective way of estimating missing values [29]

In our view, an imputation ability of one method over another seems highly problem dependent. Secondly, the more advanced methods require higher computational resources to complete their operations, which may not be justifiable given that our dataset has less than 1% MV in both the training and test sets [23]. Hence, the focus of this work is on testing the most popular MV methods. For now, we leave the use of the complex techniques in estimating missing values to future work.

III. EMPIRICAL FRAMEWORK

A. EXPERIMENTAL SETUP

The experiments are conducted using 'SPyDER' (*Scientific Python Development EnviRonment*) on the Anaconda Python distribution environment. In this paper, the aim is to combine three approaches to observe the effects this combination on anomaly detection in drinking-water quality classification problem with imbalanced class distribution and incomplete values in data. Hence, the experimental simulation is a five-way repeated-measures strategy, which allows the main effect factors (10-classifiers, 7-missing data methods, 8-resampling methods, stratified 5-Kfold cross-validation [20] and 3-performance metrics) evaluated against interaction with the random effect factor one dataset. The experiments are conducted using Intel Xeon CPU@3.20GHz, 16GB RAM system. The default settings of the examined classifiers were kept throughout the entire experiments. Fig. 1 depicts the general framework followed in conducting the experimental data analysis. The MV and ICD methods considered in this paper were selected using their popularity and citation rates as criteria. All the methods used in this

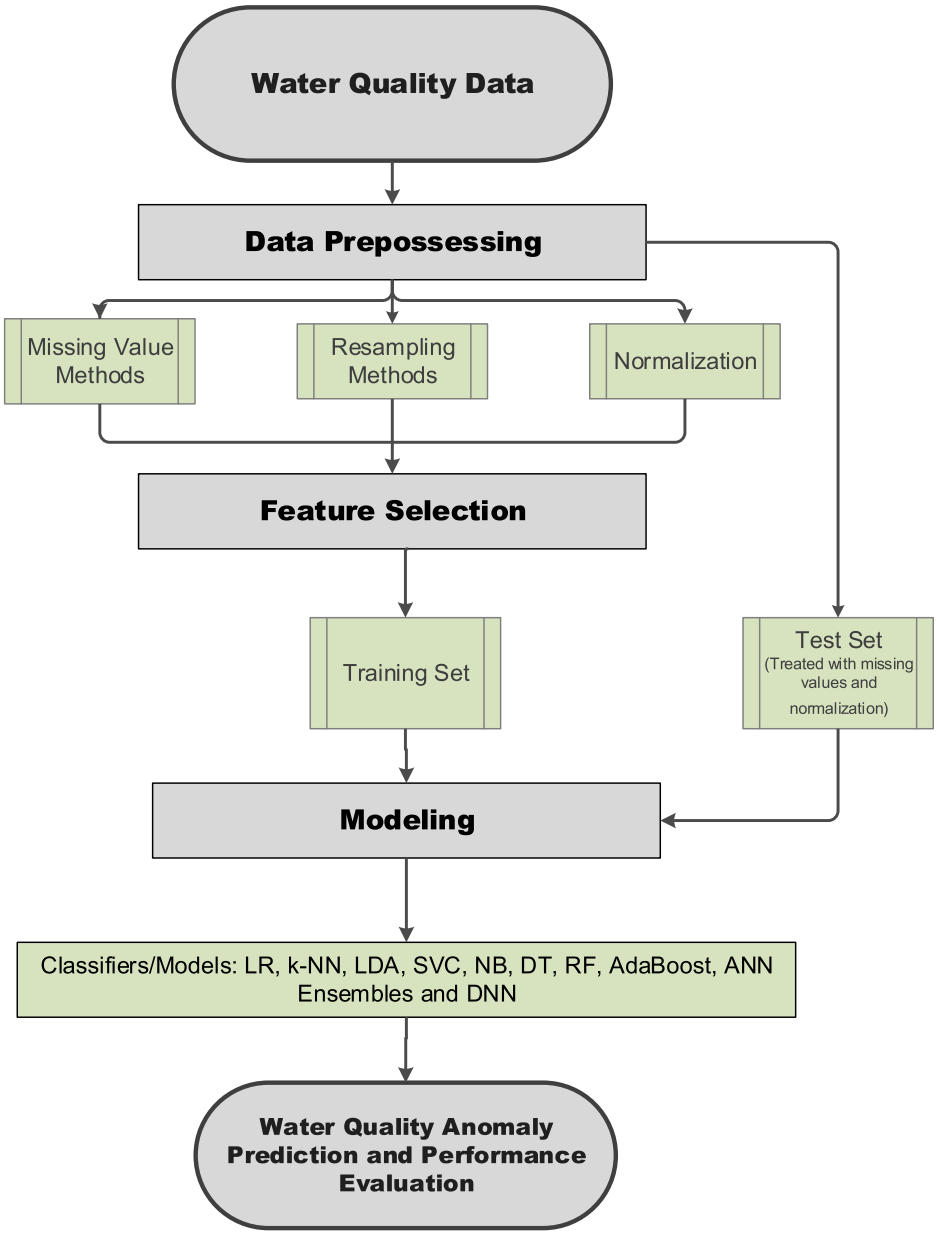


FIGURE 1. The experimental evaluation procedure.

study are listed and summarized in Table 1. They comprise of seven missing value, eight resampling methods (three oversampling, three undersampling, and two hybrids), ten classifiers, nineteen variants of homogeneous and heterogeneous ensemble methods and one optimized deep neural networks model.

B. DATASET

The dataset used in all our experiments is obtained from GECCO 2018 industrial challenge project [1], sourced from *Thüringer Fernwasserversorgung* public water utility company located in Germany. The dataset is a time series based and made up of ten independent variables, and one dependant variable. The instances have null values in all the columns except in the 'Time' and 'EVENT' columns. We assume that the dataset is missing completely at random (MCAR), which implies that the probability of the data being missing is the same for all observations, that is, there is no relationship with other data present or missing that make an observation more likely to be missing. More so on inspection of the dataset, we observe that the missing data are all within a certain range. The goal of this dataset is a classification problem intended for drinking-water quality anomaly detection, to predict if there is an event or not. The 'EVENT' is the dependant variable that is to be predicted, as either 'True' or 'False'. The training and test dataset components are summarized in Table 2. The majority of the data belongs to false majority class-0, whereas true minority class-1.

TABLE 1. Experimental study methods.

#	Method	Parameters settings	Reference
Missing values			
1.	Replace with zero (zero)	strategy = 'constant', fill_value=0	
	Listwise deletion		
2.	Mean	strategy = 'mean'	[4]
3.	Mode	Strategy = 'most_frequent'	
4.	Random Forest imputation (missForest)	N_estimators=10, criterion=entropy	[30]
5.	Expectation Maximization single imputation (EMSI)	loops=10,inplace=True	[7], [31], [32]
6.	Multiple imputations by chain equations (MICE)	Default parameters setting	[4], [33].
Data-level resampling			
1.	Random minority Undersampling (RUS)	sampling_strategy='auto'	
2.	Extraction of majority-minority Tomek links (Tomek)	sampling_strategy='auto'	[34]
3.	Repeated Edited Nearest Neighbor (RENN)	n_neighbor='3', max_iter=100	[35]
4.	Random majority Oversampling (ROS)	Default settings: sampling_strategy='auto'	
5.	Synthetic Minority Over-sampling Technique (SMOTE)	sampling_strategy='auto', k_neighbors=5	[36]
6.	Oversample using Adaptive Synthetic (ADASYN)	Default settings: n_neighbors=5	[37]
7.	Oversample with SMOTE and cleaning with Tomek (SMOTE+Tomek)	Default parameters setting	[38]
8.	Oversample with SMOTE and cleaning with ENN (SMOTE+ENN)	Default parameters setting	[39]
Classifiers			
1.	Logistic Regression (LR)	Default parameters setting	[40]
2.	K-Nearest Neighbors (k-NN)	n_neighbors=5	[41]
3.	Linear Discriminant Analysis (LDA)	Default parameters setting	[21]
4.	Support vector machine	Kernel=rbf, gamma=scale, probability=True,	[42]
5.	Naïve Bayes (NB)	Default settings	[43], [44]
6.	Decision Trees (DT)	criterion=entropy	[45]
7.	Random Forest (RF)	n_estimators=10, criterion=entropy	[46]
8.	AdaBoost (ABC)	n_estimators=50, estimator=decision tree	[47]
9.	Artificial Neural Network (ANN)	3-layers (input, hidden, output), optimizer=nadam, # neurons=6, epoch=10, batch_size=100	[48]
10.	Deep Neural Network (DNN)	5-layers, optimizer=nadam, neurons=6, epoch=10, batch_size=100	[48]
Cross-validation			
1.	Stratified K-fold	N_splits=5, shuffle=True, random_state=1	[20]
Ensemble (voting=hard scheme)			
1.	Ensemble model-1	Classifiers: k-nn, svc & rfc	[49]
2.	Ensemble model-2	Classifiers: k-nn, svc & dtc	
3.	Ensemble model-3	Classifiers: k-nn, svc & abc	
4.	Ensemble model-4	Classifier: k-nn, dtc, rfc	
5.	Ensemble model-5	Classifiers: k-nn, svc, dtc & rfc	
6.	Ensemble model-6	Classifiers: k-nn, svc, abc, dtc & rfc	
Ensemble (voting=soft scheme)			
1.	Ensemble model-7	Classifiers: k-nn, svc & rfc	[49]
2.	Ensemble model-8	Classifiers: k-nn, svc & dtc	
3.	Ensemble model-9	Classifiers: k-nn, svc & abc	
4.	Ensemble model-10	Classifier: k-nn, dtc, rfc	
5.	Ensemble model-11	Classifiers: k-nn, svc, dtc & rfc	
6.	Ensemble model-12	Classifiers: k-nn, svc, abc, dtc & rfc	
Homogeneous and heterogeneous Ensembles			
1	EasyEnsemble	Default settings: the ensemble of AdaBoost with RUS	[50]
2.	RUSBoost	Default settings: DT and RUS	[51]
3.	BalancedRandomForest	Default settings: RF and RUS	[52]
4.	BalancedBagging	Default settings: 10 number DT and RUS	[53], [54]
5.	HistGradientBoosting	Default settings	[55]
6.	StackingClassifier	Classifiers: rf, rbf kernel svm and dtc, meta-classifier=lr	[56], [57]
7.	StackingClassifier	Classifiers: k-nn, rf, dtc, rbf kernel svm, meta-classifier=lr	
Optimized DNN			
1.	DNN (missForest + SMOTEENN)	Grid search optimized parameters: 4-layers, epoch=10, batch_size=100, dropout=0, #neuron=10, optimizer=nadam	This study

TABLE 2. Summary of training and test dataset characteristics.

Dataset	Instances	Majority class	Minority class	Features	Class	Missing values	Imbalance ratio (Majority/Minority)
Training set	139566	137840	1726	10	2	1044	79.86
Test set	139566	137237	2329	10	2	24480	58.93

The data was collected continuously for over 98 days between 03/08/2016 and 13/02/2017 at an interval of 60 seconds in between readings. The dataset has a time series variable that was not included in this current study for two reasons. Firstly, the goal of this paper is to investigate the mitigating effects of MV and ICD on learning classifiers. Secondly, the time-series analysis on this dataset has been addressed in previous studies such as in [15], [16].

C. PERFORMANCE METRICS FOR IMBALANCED CLASS PROBLEM

Performance metrics for comparing experimental results in imbalanced classification problems are fundamental in discovering the quality of relationships between the data and the predicted event targets [58], [59]. In other words, the performance metric aims to show how well a learning algorithm can predict given some data observations. The most commonly accepted performance evaluation methods in imbalance classification problems are Accuracy, Sensitivity, Specificity, Precision, Recall, Balanced accuracy, ROC-AUC and F1-measure, G-mean and Matthews Correlation Coefficient [12]. However, The authors in [2] advocate using a combination of these different metrics to assess the goodness of fit for models better, as reliance on only one metric, such as accuracy may be misleading, as accuracy metric is biased toward the majority class. Although there have been studies on strategies for selecting performance metrics used for evaluating classifiers in an imbalanced scenario, such as in [59], [60], this is, however, beyond the scope of this study. Hence, the performance metrics, ROC-AUC and F1-measure, considered in this study have been selected based on their wide adoption in imbalanced classification problems, including previous studies on water quality anomaly detection, because they take into consideration class distribution. In our own opinion, this approach will provide a fair comparison with earlier studies in this domain.

IV. EXPERIMENTAL RESULTS

A. EXPERIMENT 1: COMPARISON OF MISSING DATA METHODS

This experiment aims to compare the effect of 7 missing data methods on 10 different machine learning models on the training set. The observations would allow us to verify our hypothesis that missing values harm performance measures of learning algorithms. In our case, with the training dataset having less than 1% MV (see Table 2 above). The examined ML models were evaluated on the training data, our intuition is that this gives a reasonable estimation of the performance of the different ML algorithms on the future test set (unseen data). Table 3 presents the results when applying only the MV methods namely replacing missing values with zero, list-wise deletion, mean, mode, missForest, EM, and MICE. The result also provides a robust estimate of the performance of the different machine learning algorithms on the imbalanced training dataset. The graphical results are shown in Fig. 12 in Appendix A. Fig. 2 shows the data distribution before applying the imbalanced methods, with the majority class-0 labels (False) in red colour outnumbering the minority class-1 labels (True) in blue colour.

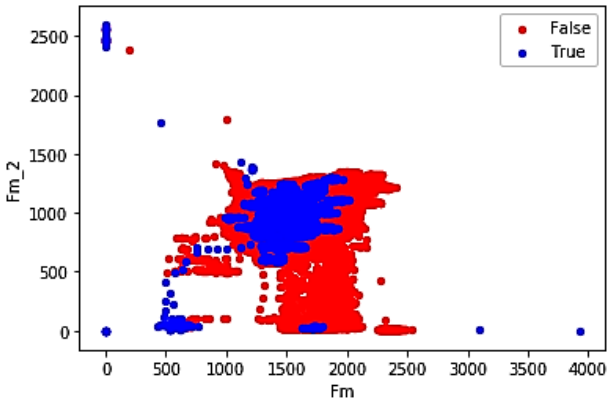


FIGURE 2. Data before applying resampling methods [(0, 275077), (1, 4055)].

Additionally, based on the results obtained in Table 3, the classifiers were ranked in terms of F1-measure based on the MV methods applied. The statistical results are outlined in Table 10 in Appendix B. We observe that RF, DT, k-NN, SVC and AdaBoost were consistently the top-5 performers in terms of F1-measure. While the neural network models (ANN and DNN), in addition to LR, LDA and NB, were the worst performers. The reason for this can be attributed to the learning philosophies of the different learning algorithms. The results also show that the neural networks are the most sensitive to the imbalanced training set, and the high accuracy results obtained by the different ML classifiers are misleading. This observation is in line with findings in numerous works of literature. All the best performing combinations are highlighted in bold.

B. EXPERIMENT 2: COMPARISON OF MISSING VALUE IN COMBINATION WITH RESAMPLING METHODS

This experiment aims to compare the effect of the combination of 7 missing data and 8 resampling methods on the 10 machine learning models. The observations would allow

TABLE 3. Results for evaluating classifiers with missing values but no resampling (stratified 5-Kfold CV).

Model	Metric		
	Balanced accuracy	ROC AUC	F1-measure
	Filling with a value (value=0)		
LR	0.567 (+/- 0.015)	0.708 (+/- 0.030)	0.236 (+/- 0.046)
k-NN	0.927 (+/- 0.015)	0.978 (+/- 0.008)	0.909 (+/- 0.013)
LDA	0.640 (+/- 0.028)	0.683 (+/- 0.040)	0.433 (+/- 0.068)
SVC	0.671 (+/- 0.025)	0.887 (+/- 0.023)	0.507 (+/- 0.053)
NB	0.538 (+/- 0.007)	0.577 (+/- 0.057)	0.135 (+/- 0.020)
DT	0.973 (+/- 0.006)	0.973 (+/- 0.006)	0.957 (+/- 0.009)
RF	0.970 (+/- 0.005)	0.991 (+/- 0.005)	**0.966 (+/- 0.009)**
AdaBoost	0.796 (+/- 0.027)	0.986 (+/- 0.001)	0.702 (+/- 0.050)
Simple ANN	0.993 (+/- 0.003)	0.903 (+/- 0.034)	0.226 (+/- 0.026)
DNN	0.995 (+/- 0.001	0.947 (+/- 0.019)	0.281 (+- 0.018)
	Listwise deletion		
LR	0.621 (+/- 0.026)	0.756 (+/- 0.033)	0.388 (+/- 0.066)
k-NN	0.958 (+/- 0.012)	0.988 (+/- 0.004)	0.952 (+/- 0.013
LDA	0.612 (+/- 0.031)	0.701 (+/- 0.031)	0.366 (+/- 0.083)
SVC	0.940 (+/- 0.015)	0.991 (+/- 0.005))	0.932 (+/- 0.015)
NB	0.729 (+/- 0.009)	0.939 (+/- 0.006)	0.484 (+/- 0.030)
DT	0.977 (+/- 0.006)	0.977 (+/- 0.006)	0.957 (+/- 0.003)
RF	0.973 (+/- 0.008)	0.991 (+/- 0.005)	**0.969 (+/- 0.007)**
AdaBoost	0.820 (+/- 0.024)	0.985 (+/- 0.003)	0.731 (+/- 0.048)
Simple ANN	0.995 (+/- 0.001)	0.970 (+/- 0.015)	0.302 (+/- 0.017)
DNN	0.997 (+/- 0.000)	0.972 (+/- 0.12)	0.346 (+- 0.011)
	Mean		
LR	0.611 (+/- 0.027)	0.751 (+/- 0.011)	0.362 (+/- 0.072)
k-NN	0.957 (+/- 0.014)	0.989 (+/- 0.008)	0.949 (+/- 0.021)
LDA	0.613 (+/- 0.024)	0.689 (+/- 0.027)	0.367 (+/- 0.063)
SVC	0.941 (+/- 0.017)	0.992 (+/- 0.005)	0.933 (+/- 0.018)
NB	0.738 (+/- 0.032)	0.944 (+/- 0.012)	0.469 (+/- 0.042)
DT	0.973 (+/- 0.007)	0.973 (+/- 0.007)	0.956 (+/- 0.010)
RF	0.972 (+/- 0.009)	0.992 (+/- 0.006)	**0.968 (+/- 0.012)**
AdaBoost	0.813 (+/- 0.022)	0.986 (+/- 0.003)	0.720 (+/- 0.012)
Simple ANN	0.996 (+/- 0.003)	0.971 (+/- 0.015))	0.228 (+/- 0.028)
DNN	0.997 (+/- 0.001)	0.972 (+/- 0.103)	0.323 (+- 0.012)
	Mode		
LR	0.602 (+/- 0.019)	0.645 (+/- 0.031)	0.337 (+/- 0.054)
k-NN	0.957 (+/- 0.016)	0.989 (+/- 0.008)	0.949 (+/- 0.022)
LDA	0.610 (+/- 0.015)	0.602 (+/- 0.051)	0.357 (+/- 0.042)
SVC	0.940 (+/- 0.018)	0.991 (+/- 0.005)	0.933 (+/- 0.020)
NB	0.733 (+/- 0.032)	0.941 (+/- 0.012)	0.461 (+/- 0.041)
DT	0.972 (+/- 0.010)	0.972 (+/- 0.010)	0.956 (+/- 0.008)
RF	0.970 (+/- 0.006)	0.992 (+/- 0.005)	**0.966 (+/- 0.009)**
AdaBoost	0.814 (+/- 0.026)	0.986 (+/- 0.002)	0.716 (+/- 0.045)
Simple ANN	0.996 (+/- 0.003)	0.965 (+/- 0.016)	0.272 (+/- 0.020)
DNN	0.997 (+/- 0.000)	0.973 (+/- 0.010)	0.334 (+- 0.011)
	missForest		
LR	0.611 (+/- 0.027)	0.766 (+/- 0.014)	0.362 (+/- 0.071)
k-NN	0.957 (+/- 0.015)	0.989 (+/- 0.008)	0.949 (+/- 0.021)
LDA	0.614 (+/- 0.024)	0.706 (+/- 0.025)	0.369 (+/- 0.063)
SVC	0.941 (+/- 0.017)	0.991 (+/- 0.005)	0.933 (+/- 0.018)
NB	0.738 (+/- 0.031)	0.944 (+/- 0.012)	0.469 (+/- 0.043)
DT	0.973 (+/- 0.005)	0.973 (+/- 0.005)	0.957 (+/- 0.009)
RF	0.970 (+/- 0.007)	0.991 (+/- 0.006)	**0.966 (+/- 0.009)**
AdaBoost	0.810 (+/- 0.037)	0.986 (+/- 0.002)	0.707 (+/- 0.052)
Simple ANN	0.995 (+/- 0.002)	0.941 (+/- 0.023)	0.290 (+/- 0.018)
DNN	0.997 (+/- 0.001)	0.975 (+/- 0.010)	0.338 (+/- 0.009)
	EM		
LR	0.581 (+/- 0.019)	0.717 (+/- 0.021)	0.278 (+/- 0.057)
k-NN	0.954 (+/- 0.016)	0.988 (+/- 0.007)	0.900 (+/- 0.015)
LDA	0.603 (+/- 0.017)	0.644 (+/- 0.040)	0.324 (+/- 0.047)
SVC	0.865 (+/- 0.028)	0.986 (+/- 0.002)	0.825 (+/- 0.031)
NB	0.684 (+/- 0.035)	0.922 (+/- 0.014)	0.375 (+/- 0.048)
DT	0.961 (+/- 0.011)	0.961 (+/- 0.011)	0.914 (+/- 0.024)
RF	0.954 (+/- 0.013)	0.992 (+/- 0.006)	**0.948 (+/- 0.016)**
AdaBoost	0.701 (+/- 0.032)	0.972 (+/- 0.004)	0.516 (+/- 0.057)
Simple ANN	0.990 (+/- 0.001)	0.934 (+/- 0.033)	0.193 (+/- 0.296)
DNN	0.993 (+/- 0.001)	0.962 (+/- 0.024)	0.227 (+/- 0.025)

TABLE 3. *(Continued.)* **Results for evaluating classifiers with missing values but no resampling (stratified 5-Kfold CV).**

	MICE		
LR	0.611 (+/- 0.027)	0.752 (+/- 0.013)	0.362 (+/- 0.072)
k-NN	0.957 (+/- 0.014)	0.989 (+/- 0.008)	0.949 (+/- 0.021)
LDA	0.613 (+/- 0.024)	0.691 (+/- 0.027)	0.367 (+/- 0.063)
SVC	0.941 (+/- 0.017)	0.992 (+/- 0.005)	0.933 (+/- 0.018)
NB	0.738 (+/- 0.032)	0.944 (+/- 0.012)	0.469 (+/- 0.042)
DT	0.974 (+/- 0.007)	0.974 (+/- 0.007)	0.956 (+/- 0.010)
RF	0.972 (+/- 0.008)	0.992 (+/- 0.005)	**0.968 (+/- 0.011)**
AdaBoost	0.815 (+/- 0.019)	0.986 (+/- 0.003)	0.717 (+/- 0.016)
Simple ANN	0.995 (+/- 0.003)	0.963 (+/- 0.012)	0.294 (+/- 0.015)
DNN	0.997 (+/- 0.001)	0.976 (+/- 0/012)	0.330 (+/- 0.012)

us to verify our hypothesis that class-imbalances would harm performance measures of learning algorithms. In our case, with the training set having a majority to minority ratio of 80:1 (see Table 2 and Fig. 2). Table 4 presents the results, of combining MV and resampling methods to estimate the performance of the different machine learning algorithms. While Fig. 3 shows the visual distribution maps of the training dataset with the application of the different resampling methods. It is observed that the relationship between the two classes overlaps significantly, which is one factor in an imbalance dataset scenario that not only affects the performance of learning classifiers but adds complexity to the learning algorithms in terms of for example further oversampling the training set [61]. In this experiment, SMOTE + ENN combined with all the MV methods exhibited a better performance across all the learning models in terms of F1-measure with low statistical variances. SMOTE + Tomek coming a close second best also across all the learning models in terms of F1-measure. ROS also exhibited good performance; however, the recorded F1-measure across a combination of various methods had higher statistical variances when compared to SMOTE + ENN, this in addition to the possibility of exacerbating the class overlap issue on the classifiers' performance [61]. Generally, we observe that all the undersampling methods (RUS, Tomek link and RENN) performed the worst across all combination of methods, indicating their unsuitability for this dataset. In the experiments reported in Table 4, MV and resampling methods were both applied to the training set.

C. EXPERIMENT 3: COMPARISON OF ENSEMBLE AND DNN MODELS WITH MISSFOREST AND SMOTE + ENN

Despite the significant progress achieved in machine learning research, conventional machine learning methods may not achieve satisfactory performance when dealing with imbalanced data with missing values. This is because of the inability of these methods to cope with the imbalanced dataset and their assumptions of balanced class distribution. Several studies using WQAD data and traditional machine learning algorithms have previously been conducted, but challenges associated with missing values and class-imbalance leave room for improvement. Ensemble learning and DNN have proven to be efficient approaches to dealing with imbalanced dataset problems over traditional individual classification models [62], [63].

Recently, deep learning method evaluated in combination with appropriate data preprocessing techniques (missing value and resampling method) has shown promise in improving the predictive its predictive performance [63], as well as in other water quality anomaly detection studies. Motivated by these previous findings, we proposed and employed a deep neural network in combination with missing value and resampling preprocessing methods to determine its effectiveness in addressing class-imbalance with missing values, in comparison with the ensemble methods.

This section presents the results obtained using ensemble approaches where we considered the top-5 performing pool of machine learning heterogeneous classifiers obtained in experiments 1 and 2 namely RF, DT, k-NN, SVC and AdaBoost (except for the neural network model which was evaluated separately). The top-5 machine learning classifiers were used to implement the various variations of ensemble voting-based bagging, boosting and stacking, models, which were then compared to the optimized DNN model.

We used grid search to find the best hyperprameters of the optimized DNN using F1-measure to choose the best model. We tested the following hyperparameters: neurons = [10, 20, 30, 40, 50], dropout_rate = [0.0, 0.1, 0.2, 0.3, 0.4], optimizer = ['rmsprop', 'adam', 'nadam'], epoch = [10, 50, 100] and batch_size = [20, 50, 100]. The best DNN was implemented with the following parameters: 4-layers (1-input, 2-hidden, 1-output layer), neurons = 10, activation = relu for input and hidden layers, activation = sigmoid for output layer, dropout_rate = 0.0, optimizer = nadam, loss = binary_crossentropy, epoch = 10 and batch_size = 100. The final optimized DNN architectural model used is depicted in Figure 4.

The experimental results are shown in Tables 5-7, while that of the optimized DNN is shown in Table 8. In all the experiments, MissForest MV and SMOTE + ENN resampling methods were applied to the training set due to their better performance observed in experiment 1 and 2. The pictorial representation of the balanced training data using the SMOTE + ENN method is shown in Fig. 5. The models were all evaluated on the imbalanced test set, using accuracy, ROC-AUC and F1-measure as the performance evaluation

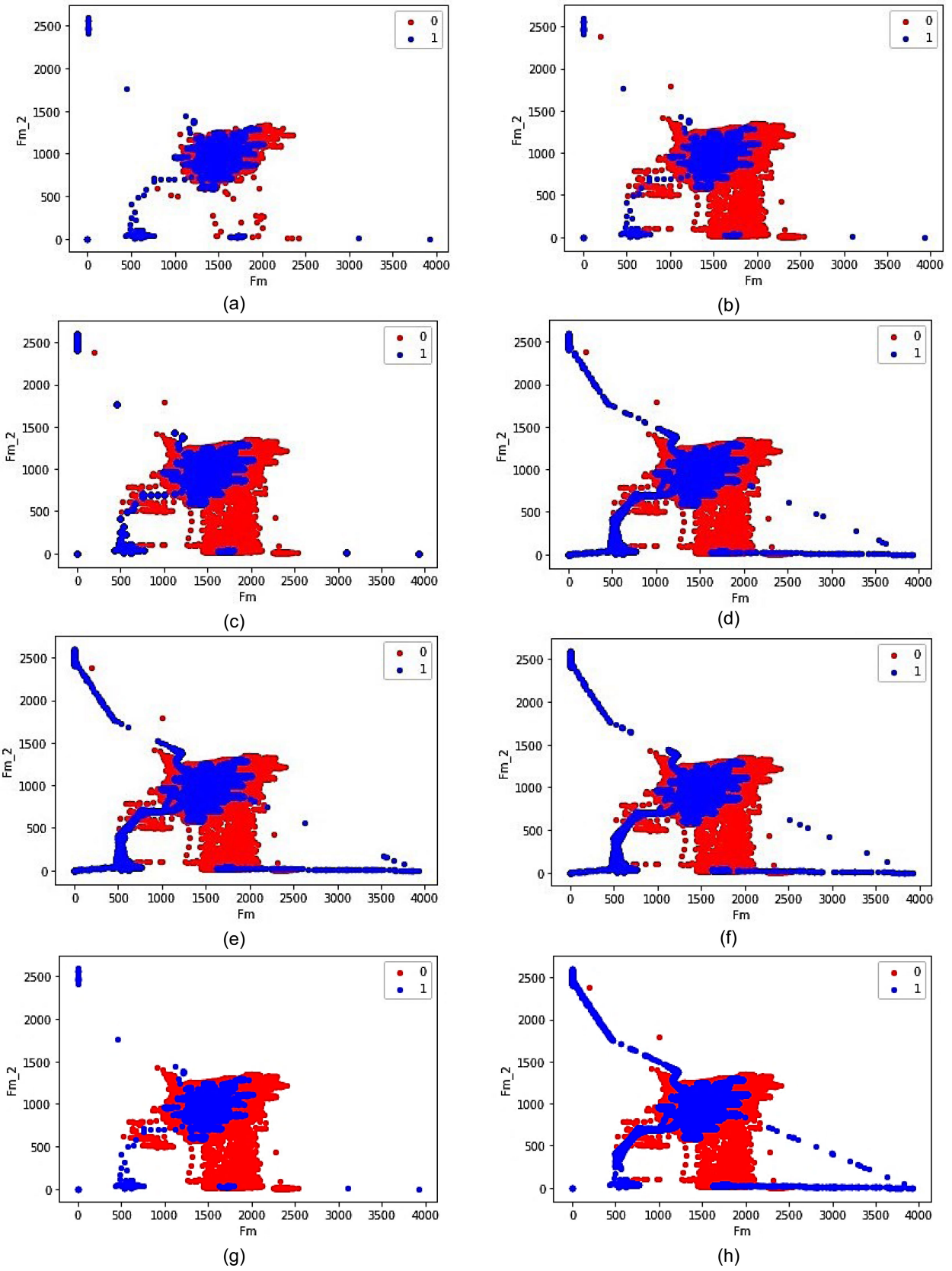


FIGURE 3. Visual training data distribution maps with the application of different resampling methods. (a) Random US [(0, 1726), (1, 1726)]; (b) Tomek Links US [(0, 137564), (1, 1726), (c) Random OS [(0.0, 137840), (1.0, 137840)], (d) SMOTE OS [(0.0, 137840), (1.0, 137840)]; (e) SMOTE + Tomek [(0, 137736), (1, 137736); (f) SMOTE + ENN [(0, 241687), (1, 246574), (g) RENN US [(0, 136330), (1, 1726)]; (h) ADASYN OS [(0, 137840), (1, 137912)].

TABLE 4. F1-measure results with missing value and resampling methods (stratified 5-Kfold CV).

Methods	Oversampling			Under-sampling			Hybrid sampling	
	ROS	SMOTE	ADASYN	RUS	Tomek links	RENN	SMOTE + Tomek	SMOTE + ENN
Filling with a value (value=0)								
LR	0.6704	0.6817	0.4208	0.6139	0.4079	0.4057	0.6812	0.6848
k-NN	0.9285	0.8979	0.8538	0.6837	0.3148	0.3314	0.8971	0.9029
LDA	0.5831	0.5901	0.4718	0.4720	0.4131	0.4149	0.5926	0.5988
SVC	0.9511	0.9361	0.8075	0.8029	0.3936	0.4083	0.9360	0.9349
NB	0.5787	0.5273	0.5527	0.5051	0.2045	0.2056	0.5255	0.5250
DT	0.9442	0.9089	0.8301	0.5919	0.2830	0.3153	0.8884	0.8888
RF	0.9630	0.9461	0.8757	0.6958	0.3762	0.3432	0.9228	0.9296
AdaBoost	0.8996	0.9207	0.7538	0.8074	0.4467	0.4114	0.9143	0.9083
Simple ANN	0.9844	0.9716	0.9515	0.7424	0.6033	0.6391	0.9689	0.9721
DNN	**0.9899**	**0.9801**	**0.9704**	**0.8899**	**0.6582**	**0.6520**	**0.9811**	**0.9837**
Listwise deletion								
LR	0.7113	0.7114	0.2698	0.6638	0.3595	0.3521	0.7116	0.7149
k-NN	0.9319	0.8983	0.8728	0.6576	0.3893	0.4070	0.8982	0.9048
LDA	0.5748	0.5853	0.4410	0.5287	0.4276	0.4277	0.5823	0.5883
SVC	0.9459	0.9299	0.8268	0.8064	0.6463	0.6650	0.9296	0.9307
NB	0.8646	0.8307	0.7844	0.8649	0.5124	0.5182	0.8294	0.8298
DT	0.9423	0.9027	0.8293	0.4880	0.3056	0.2709	0.8865	0.8911
RF	0.9682	0.9682	0.8656	0.6497	0.3866	0.4222	0.9205	0.9272
AdaBoost	0.8985	0.9333	0.8030	0.7878	0.4562	0.3766	0.9277	0.9195
Simple ANN	0.9845	0.9753	0.9755	0.7213	0.6446	0.6723	0.9716	0.9709
DNN	**0.9913**	**0.9828**	**0.9803**	**0.9624**	**0.6646**	**0.6795**	**0.9851**	**0.9837**
Mean								
LR	0.7119	0.7128	0.2710	0.6672	0.3592	0.3513	0.7116	0.7152
k-NN	0.9298	0.8969	0.8716	0.6827	0.3888	0.4070	0.8972	0.9028
LDA	0.5754	0.5853	0.4425	0.5359	0.4288	0.4283	0.5829	0.5892
SVC	0.9454	0.9360	0.8220	0.8073	0.6480	0.6645	0.9354	0.9360
NB	0.8654	0.8293	0.7856	0.8684	0.5118	0.5186	0.8305	0.8308
DT	0.9450	0.9051	0.8283	0.6036	0.2502	0.2714	0.8933	0.8906
RF	0.9666	0.9359	0.8642	0.8313	0.3816	0.3906	0.9223	0.9241
AdaBoost	0.8898	0.9430	0.7940	0.8466	0.4424	0.4869	0.9363	0.9351
Simple ANN	0.9855	0.9761	0.9700	0.8381	0.6468	0.6729	0.9744	0.9738
DNN	**0.9919**	**0.9851**	**0.9816**	**0.9710**	**0.6944**	**0.6646**	**0.9814**	**0.9875**
Mode								
LR	0.7110	0.7126	0.2704	0.6666	0.3674	0.3616	0.7116	0.7151
k-NN	0.9298	0.8971	0.8717	0.6827	0.3890	0.4080	0.8972	0.9027
LDA	0.5743	0.5829	0.4439	0.5362	0.4275	0.4267	0.5823	0.5885
SVC	0.9443	0.9329	0.8212	0.8064	0.6481	0.6648	0.9328	0.9334
NB	0.8651	0.8308	0.7854	0.8684	0.5118	0.5187	0.8303	0.8306
DT	0.9555	0.8993	0.8084	0.6036	0.2503	0.2711	0.9127	0.8946
RF	0.9698	0.9320	0.8670	0.8314	0.3918	0.3788	0.9230	0.9125
AdaBoost	0.9010	0.9300	0.7954	0.8542	0.3618	0.3854	0.9387	0.9293
Simple ANN	0.9829	0.9704	0.9730	0.8011	0.6344	0.6615	0.9749	0.9785
DNN	**0.9907**	**0.9841**	**0.9777**	**0.9762**	**0.6980**	**0.6850**	**0.9845**	**0.9850**
missForest								
LR	0.7124	0.7119	0.2678	0.6677	0.3595	0.3521	0.7118	0.7153
k-NN	0.9300	0.8972	0.8729	0.6827	0.3889	0.4079	0.8976	0.9032
LDA	0.5735	0.5810	0.4412	0.5366	0.4269	0.4273	0.5807	0.5870
SVC	0.9444	0.9313	0.8201	0.8068	0.6476	0.6648	0.9306	0.9314
NB	0.8656	0.8301	0.7876	0.8681	0.5119	0.5182	0.8304	0.8307
DT	0.9475	0.8996	0.8567	0.6036	0.2502	0.2736	0.8886	0.8892
RF	0.9677	0.9306	0.8472	0.8321	0.3872	0.3875	0.9355	0.9338
AdaBoost	0.9033	0.9380	0.7967	0.8539	0.4118	0.3973	0.9371	0.9328
Simple ANN	0.9874	0.9787	0.9722	0.8726	0.6268	0.6785	0.9714	0.9545
DNN	**0.9902**	**0.9848**	**0.9796**	**0.9712**	**0.6637**	**0.6721**	**0.9812**	**0.9844**
EM								
LR	0.7105	0.7103	0.3485	0.6657	0.3771	0.3763	0.7103	0.7141
k-NN	0.9389	0.9068	0.8848	0.6719	0.3887	0.4059	0.9065	0.9094
LDA	0.5742	0.5819	0.4452	0.5364	0.4240	0.4252	0.5821	0.5886
SVC	0.9497	0.9374	0.8254	0.8060	0.6080	0.6083	0.9361	0.9366
NB	0.8614	0.8266	0.7807	0.7800	0.5046	0.5117	0.8265	0.8268
DT	0.9531	0.9243	0.8359	0.6399	0.3527	0.3034	0.9124	0.9205
RF	0.9799	0.9652	0.8818	0.7067	0.7067	0.3740	0.9587	0.9570
AdaBoost	0.9286	0.9417	0.8163	0.8202	0.5028	0.4569	0.9491	0.9419
Simple ANN	0.9873	0.9699	0.9635	0.7497	0.6426	0.6312	0.9774	0.9731

TABLE 4. (*Continued.*) F1-measure results with missing value and resampling methods (stratified 5-Kfold CV).

DNN	**0.9893**	**0.9836**	**0.9798**	**0.9696**	**0.6135**	**0.6668**	**0.9848**	**0.9841**
				MICE				
LR	0.7119	0.7115	0.2707	0.6669	0.3592	0.3513	0.7116	0.7152
k-NN	0.9298	0.8977	0.8731	0.6827	0.3888	0.4070	0.8972	0.9028
LDA	0.5754	0.5822	0.4415	0.5357	0.4286	0.4281	0.5829	0.5892
SVC	0.9449	0.9353	0.8183	0.8077	0.6479	0.6648	0.9350	0.9356
NB	0.8654	0.8306	0.7864	0.8684	0.5118	0.5186	0.8304	0.8308
DT	0.9449	0.9065	0.8361	0.6036	0.2503	0.2715	0.8933	0.8922
RF	0.9658	0.9319	0.8721	0.8398	0.3858	0.3905	0.9270	0.9206
AdaBoost	0.8898	0.9434	0.8015	0.8898	0.4537	0.4869	0.9372	0.9209
Simple ANN	0.9849	0.9759	0.9701	0.8966	0.6436	0.6630	0.9773	0.9732
DNN	**0.9892**	**0.9829**	**0.9758**	**0.9343**	**0.6560**	**0.6809**	**0.9819**	**0.9808**

TABLE 5. Results for 6 variations of heterogeneous ensemble approaches (voting = hard scheme).

Ensemble model	Metric		
	Balanced accuracy	ROC_AUC	F1-measure
Ensemble 1 (k-nn, svc & rfc)	0.609 (+/- 0.067)	0.603 (+/- 0.067)	0.239 (+/- 0.221)
Ensemble 2 (k-nn, svc & dtc)	0.609 (+/- 0.063)	0.609 (+/- 0.063)	0.241 (+/- 0.218)
Ensemble 3 (k-nn, svc &abc)	0.616 (+/- 0.050)	0.616 (+/- 0.050)	0.257 (+/- 0.191)
Ensemble 4 (k-nn, dtc, rfc)	0.696 (+/- 0.084)	0.347 (+/- 0.264)	0.347 (+/- 0.264)
Ensemble 5 (k-nn, svc, dtc & rfc)	0.610 (+/- 0.062)	0.610 (+/- 0.062)	0.254 (+/- 0.226)
Ensemble 6 (k-nn, svc, abc, dtc & rfc)	0.668 (+/- 0.077)	0.368 (+/- 0.276)	**0.368 (+/- 0.276)**

TABLE 6. Results for 6 variations of heterogeneous ensemble approaches (voting = soft scheme).

Ensemble model	Metric		
	Balanced accuracy	ROC_AUC	F1-measure
Ensemble 7 (k-nn, svc & rfc)	0.623 (+/- 0.094)	0.623 (+/- 0.094)	0.194 (+/- 0.254)
Ensemble 8 (k-nn, svc & dtc)	0.308 (+/- 0.230)	0.308 (+/- 0.230)	0.308 (+/- 0.230)
Ensemble 9 (k-nn, svc & abc)	0.134 (+/- 0.170)	0.134 (+/- 0.170)	0.134 (+/- 0.170)
Ensemble 10 (k-nn, dtc, rfc)	0.698 (+/- 0.083)	0.350 (+/- 0.264)	**0.350 (+/- 0.264)**
Ensemble 11 (k-nn, svc, dtc & rfc)	0.346 (+/- 0.258)	0.346 (+/- 0.258)	0.346 (+/- 0.258)
Ensemble 12 (k-nn, svc, abc, dtc & rfc)	0.347 (+/- 0.258)	0.347 (+/- 0.258)	0.347 (+/- 0.258)

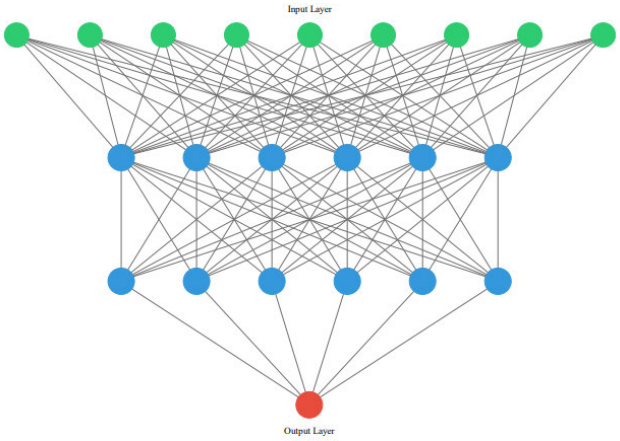


FIGURE 4. The optimized DNN model architecture.

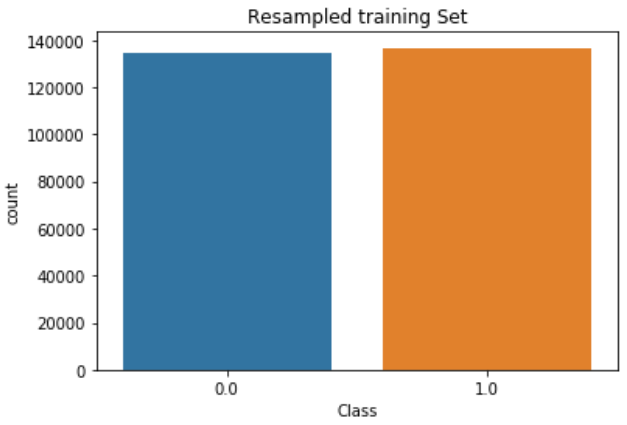


FIGURE 5. Balanced training set using SMOTE + ENN method [(0, 134423), (1, 136749)].

metrics. We observe that even though the classifiers benefited from the combination of MV and ICD in the training set, they suffered significantly when evaluated on the imbalance test set in terms of the performance measures. This is attributed to the sensitive of classifiers to the level of imbalance in the presence of classes' overlap, which leads to difficulty in distinguishing between the two classes because of nearly equal prior probabilities estimates of both classes [12].

D. STATISTICAL TEST

In this subsection, we apply statistical tests to evaluate and validate the results obtained from the experiments conducted.

TABLE 7. Results for different variations of boosting, bagging and stacking ensemble approaches.

Model	Metric		
	Balance accuracy	ROC_AUC	F1-measure
EasyEnsemble Classifier (EE)	0.763 (+/- 0.234)	0.763 (+/- 0.117)	0.279 (+/- 0.446)
RUSBoost Classifier (RUSB)	0.559 (+/- 0.014)	0.559 (+/- 0.007)	0.107 (+/- 0.113)
BalancedRandomForest Classifier (BRF)	0.734 (+/- 0.193)	0.327 (+/- 0.287)	0.327 (+/- 0.575)
BalancedBagging Classifier (BB)	0.678 (+/- 0.156)	0.678 (+/- 0.078)	0.288 (+/- 0.452)
HistGradientBoosting Classifier (HGB)	0.648 (+/- 0.188)	0.771 (+/- 0.192)	0.285 (+/- 0.465)
Stacking Classifier (rfc, rbf kernel svm and dtc) (STK1)	0.691 (+/- 0.111)	0.691 (+/- 0.111)	**0.359 (+/- 0.258)**
Stacking Classifier (k-nn, rf, dtc, rbf kernel svm) (STK2)	0.696 (+/- 0.084)	0.352 (+/- 0.264)	0.352 (+/- 0.264)

TABLE 8. Result for the optimized deep neural network.

Neural Network Model	Metric		
	Balanced accuracy	ROC AUC	F1-measure
DNN (missForest+SMOTEENN)	0.983	0.667	**0.410**

We have evaluated 10 classifiers on 7 missing values and 8 resampling methods, using F1-measure as the main performance score.

Recall the results obtained in experiment 1 presented in Table 3, ten learning algorithms were evaluated using seven different missing value methods. Similarly, for the results in experiment 2 presented in Table 4, ten learning algorithms were evaluated using seven different missing value and eight resampling methods. The evaluations were based on three performance metrics namely accuracy, ROC-AUC and F1-measure. Lastly, in the results from experiment 3 presented in Tables 5 - 7, the top-5 machine learning classifiers in experiment 2 were used to implement the various variations of voting-based ensemble models, which are then compared to the bagging, boosting and stacking ensemble models. Based on the performance of DNN in experiment 2, we went ahead to develop an optimized DNN, to verify if we could get a better result. The result for the optimized DNN is present in Table 8.

Our aim here is to find out if the performance differences between the different learning algorithms are statistically significant. To assess the results obtained for each classifier, we adopt the non-parametric Friedman test proposed in [64]. The Friedman test is firstly used to evaluate the acceptance or rejection of the null hypothesis (H_0) that all classifiers perform equally for a given significance or risk level (alpha level). Therefore, in our case, the null-hypothesis being tested is that all the classifiers performed the same and the observed differences are merely random or by coincidence.

The Friedman test ranks the algorithms for each data set separately. The best performing algorithm gets the rank of 1, the second-best rank 2 etc. In the case of ties, it assigns average ranks. Then, the Friedman test compares the average ranks of each algorithm and calculates the Friedman statistic. If a statistically significant difference in the performance is detected, we proceed with a *post hoc* test. We use the *post hoc* Nemenyi test to compare all the classifiers to each other. In this procedure, the performance of two classifiers is significantly different if their average ranks differ more than some critical distance (CD). The critical distance depends on the number of algorithms, the number of data sets, metrics and the critical value (for a given significance level p) that is based on the Studentized range statistic [64].

As recommended in [64], caution is to be applied regarding the statistical process to be used when testing multiple classifiers on a single dataset. This avoids biased estimation and Type I error because the mean performance and variance computed from the repeated training/test random samples are related. To avoid this trap, we test the ten classifiers on the seven different missing value methods and eight different resampling methods. This way, the dataset distributions are slightly different and not entirely the same. The intuition here is that the multiple dataset distributions created from the different missing value and resampling methods are used only to evaluate the performance measures. While the differences in performance over the independent missing value and resampled datasets give us the sources of variance and a sample of independent measurements. That way, the statistical test assumes the form of comparing multiple classifiers over multiple datasets. For comparing multiple random variables, a non-parametric rank-based Friedman test is recommended in [64].

According to Demsar, 2006 [58], given r_i^j be the rank of the j^{th} classifiers (K) in the i^{th} dataset (D), under this null hypothesis setting, the Friedman test statistic is formalized in (1). X_F^2 is distributed according to chi-squared distribution, with (K-1) and (K-1) (D-1) degrees of freedom. Give a large enough X_F^2 value, then the null-hypothesis that there is no difference between the classifiers can be rejected.

$$X_F^2 = \frac{12D}{K\,(K+1)}\left[\sum_{j=1}^{k} AvegR_j^2 - \frac{K\,(K+1)^2}{4}\right] \tag{1}$$

where, $AvegR_j = \frac{1}{D}\sum_{i=1}^{D} r_i^j$ is the formula used by the Friedman test to compares the average ranks of the classifiers.

According to [64], the *post hoc* Nemenyi test states that the performance of two or more classifiers is significantly

different if their corresponding average ranks differ by at least the critical difference. In other words, the *post hoc* Nemenyi test is utilised to report any significant differences between the individual classifiers. The critical difference (CD) is expressed using the formula in (2) as:

$$CD = q_\alpha \sqrt{\frac{K\,(K+1)}{6D}} \tag{2}$$

where q_α is the critical value based on the Studentized range statistic, using significance levels of $\alpha = 0.05$ and $\alpha = 0.10$; K is the total number of classifiers, and D is the number of the dataset used in the study, the resampled datasets in our case for this study.

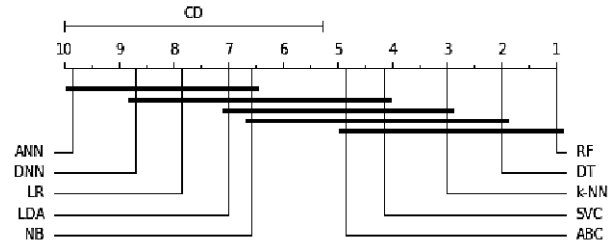


FIGURE 6. Average ranking of the ten classifiers across all the missing value methods ($\alpha = 0.1$).

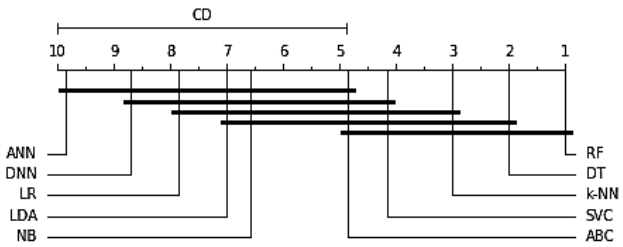


FIGURE 7. Average rankings of classifiers across all the missing value methods ($\alpha = 0.05$).

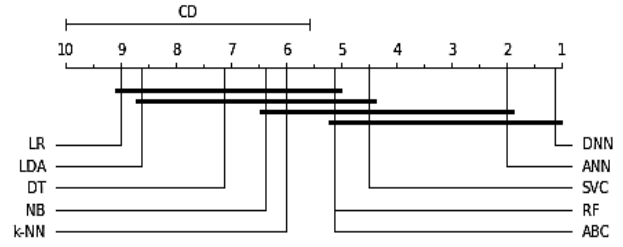


FIGURE 8. Average rankings of classifiers across all resampling methods with missForest imputation ($\alpha = 0.1$).

The average rank of classifiers for the two analyzed experimental scenarios: 1) interactions between classifiers, missing value and F1-measure, and 2) interactions between classifiers, missing values, resampling and F1-measure are shown in Table 10 in the appendix section. The average ranks are used to show the graphical representation of the *post hoc* Nemenyi test results in Figure 6-9. For experiment 3 scenario, the overall average rank of the 19 different ensemble

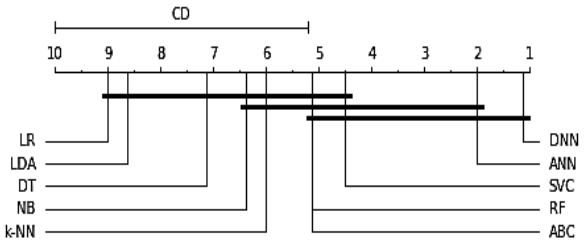


FIGURE 9. Average rankings of classifiers across all resampling methods with missForest imputation ($\alpha = 0.05$).

models and the optimized DNN model considering all three performance metrics (balanced accuracy, ROC and F-score) are shown in Table 11, also at the appendix section.

For our ten classifiers, seven missing value and eight resampling methods, df = 10-1 = 9 for classifiers, df = (10-1) ∗ (7-1) = 54 for classifiers and missing values methods and df = (10-1) ∗ (8-1) = 63 for classifiers and resampling methods. We first apply Friedman's test for the average ranking of the ten classifiers when applying only missForest imputation methods. We report the Friedman test statistic = 35.166 and a very small *p-value* = 4.002E-06. This result shows that the performances of the classifiers are statistically significantly different since the *p-value* $< \propto = 0.05$. This is an indication that the performances of the individual classifier are not equivalent. Moreover, the very small *p-value* and large enough Friedman test value obtained provides strong evidence against the null hypothesis; hence, we reject the null hypothesis. We can thus proceed to apply the *post hoc* Nemenyi test to compare any significant differences between individual classifiers.

The diagrams in Fig. 6 - 9 shows the visual representation of the average ranked performances of the ten classifiers with the critical distance of *post hoc* Nemenyi test. The results are presented using the significance level, $\alpha = 0.10$ and $\alpha = 0.05$.

In Fig. 6 the diagram shows the ranked performance of the classifiers with CD = 4.725 at a significance level, $\alpha = 0.10$, We see in Fig. 6 that ANN, DNN, LR, LDA and NB classifiers are significantly different from the best-performing classifier RF having the lowest rank across the missing value methods. While the diagram in Fig. 7 shows the average rank performance of the classifiers with CD = 5.12, at a significance level, $\alpha = 0.05$. Similarly, we observe in Fig. 7 RF is the best classifier, having the lowest rank across all the missing value methods and statistically better than ANN, DNN, LR, LDA and NB classifiers as indicated by the CD bar. In both Fig. 6 and Fig. 7, we see all the top-5 performing classifiers on the right-most part of the diagram, as against the least performing classifier to the right-most part of the diagram. The set of classifiers that do not differ significantly are grouped with a bold horizontally connected line.

Next, we perform the Friedman test on the average ranking of the ten classifiers when applying the missForest missing value method combined with the eight resampling methods,

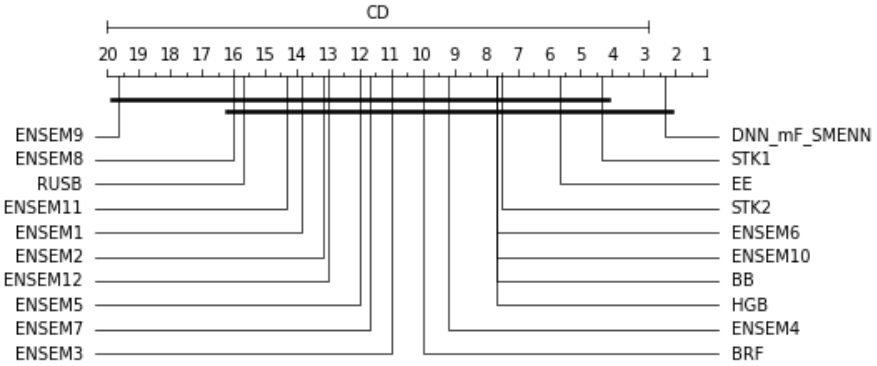


FIGURE 10. Average rankings of the ensemble and DNN models across all the performance metric (balanced accuracy, ROC and F1-score) at ($\alpha = 0.05$).

we report Friedman test statistic = 55.50 and a very small *p-value* = 1.187E-09. The result shows statistically significant different performance (*p-value* $< \alpha = 0.05$), indicating the performances of all the classifiers are not equivalent. Hence, once again, we reject the null hypothesis. Consequently, we can proceed to apply the *post hoc* Nemenyi test to compare all classifiers to each other. The diagram in Fig. 8 shows the average ranked performance of the ten classifiers along with the CD = 4.420, at a significance level, $\alpha = 0.10$. We observe in Fig. 8 that LR, LDA, DT, NB and k-NN classifiers are statistically significantly different from the best-performing classifier, this time DNN having the lowest rank across all the resampled dataset methods in combination with missForest imputation method. While the diagram in Fig. 9 shows the average ranked performance of the classifiers with CD = 4.789, at a significance level, $\alpha = 0.05$. Similarly, we observe in Fig. 9 that LR, LDA, DT, NB and k-NN classifiers are significantly different from the best-performing classifier, which is DNN also having the lowest rank across all the resampling methods with missForest imputation. In both Fig. 8 and Fig. 9, we see all the top-5 performing classifiers (DNN, ANN, SVC, RF and ABC) on the right-most part of the diagram, as against the least performing classifier (k-NN, NB, DT, LDA, and LR) to the left-most part of the diagram. It is evident in this study that RF is the least affected by imbalance data as well as showing good behaviour in comparison to the other nine classifiers. On the other hand, the neural network-based classifiers specifically DNN are the most sensitive to imbalanced class data based on the earlier results in Fig. 6 and Fig. 7. From these preliminary results, it would suggest that DNN is a worthy candidate for further study on this problem. This we achieve by investigating an optimized DNN.

Finally, we perform the statistical test on the average global ranking of the 19 ensembles and the optimized DNN models, with df = 20 - 1 = 19. This test reveals the Friedman test statistic = 24.2745; a very small, *p-value* = 5.3562E-06; at significant level $\alpha = 0.05$. The result shows statistically significant different performance, (*p-value* $< \alpha =$ 0.05) meaning the performances of all the models are not equivalent. Hence, once again, we reject the null hypothesis. Consequently, we can proceed to apply the *post hoc* Nemenyi test to compare all the 20 models to each other. The diagram in Fig. 10 shows the average rank performance of the 20 models along with CD = 17.1182, at a significance level, $\alpha = 0.05$. The result reveals that although the DNN_mF_SMENN model is consistently having a better performance than the other methods, there is, however, no significant difference at $\alpha = 0.05$ (risk level) between DNN_mF_SMENN that performed the best (lowest-ranked) and 18 ensemble models that are grouped and connected with a bold line, signifying similar performance with the DNN_mF_SMENN method. The reason why the differences in performance of these methods are not significant stems from the reason that the statistical tests we use have a conservative behavior [64]. Whereas, ENSEM9 model was the method that significantly performed worse than DNN_mF_SMENN. A similar result is also observed for $\alpha = 0.1$, with CD = 16.0334, as depicted in Fig. 11. An interesting observation on the average ranking diagrams is the stacking classifiers models, ranking above most of the ensemble and RUSB methods on the left-hand side of Fig. 10 and Fig. 11, an indication of a potentially good candidate for this dataset problem.

E. COMPUTATIONAL COMPLEXITY

In this section, we discuss the computational cost-effectiveness of our proposed DNN model in comparison to the other different models. Algorithm complexity comprises of two factors, namely time complexity and space complexity. Time complexity is the amount of time required by an algorithm to complete, which is represented by the number of computational steps that a processor would take to solve a dataset problem using any individual algorithm. The runtime of each model depends on the input parameters with a dependency function T representing the model's time computational complexity. Time computational complexity is a function that represents the dependency between input dataset size and the number of floating-point operations (FLOP) required by the algorithm to complete, which is described as $T(D)$, where D is the size of the input dataset. While space

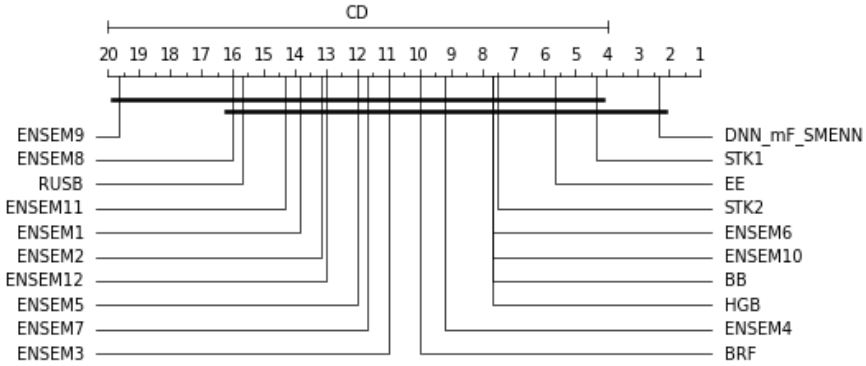


FIGURE 11. Average rankings of the ensemble and DNN models across all the performance metric (balanced accuracy, ROC and F1-score) at ($\alpha = 0.1$).

complexity is the amount of memory required by an algorithm to complete [65], [66].

In machine learning research, one of the goals of complexity evaluation is to understand how an algorithm scales when the input data size grows. In other words, complexity evaluation compares different models to know which model takes lesser resources in terms of time and space as data size (D) input grows. Big-Oh notation is a standard approach of evaluating the computational complexity or efficiency of an algorithm. Big-Oh notation [65] is a formal way of denoting an order of a function of some input based on the maximum number of operations an algorithm performs. Each machine learning model has its order of function irrespective of the number of operations. Hence, to compare models' complexities, it is sufficient to compare their Big-Oh notation represented by a constant linear dependency, which indicates how the runtime of an algorithm grows depending on the input dataset size. Big-oh notation focuses on the main computational operations while ignoring the low-level mathematical operation details [65]. For this section, we focus only on the time complexity of the learning algorithms, since more interest is usually shown to the computation speed, irrespective of the problem definition.

Evaluating the computational complexity of machine learning algorithms is no trivial endeavour. As the complexity of learning algorithms depends on many factors such as the algorithm's implementation, type of dataset problem, and other parameters passed on to the algorithm. For instance, the complex ensemble voting-based methods rely on other algorithms. We do not include the ensembles in our current analysis, however, the ensemble methods are product of the complexity of the original model by the number of voting models in the ensemble implementation. In the bagging ensemble methods, for instance, the training size is replaced by the size of each bag. Table 9 shows approximated time computational complexity of the models used in this study based on the training dataset N, the number of features P, and their specific implementations, such as the number of trees and their depth for trees based methods, number of support vectors, number of operation for comparing nearest neighbors

TABLE 9. Comparison of runtime of models on two different hardware environments.

Model	Time computational complexity/hardware environment (seconds)			
	PC (Intel® Core(TM) i3-3217U CPU @ 1.80Hz, 6GB RAM	Kaggle's cloud platform 4CPU core 16GB RAM	Difference	%
LR	2.82	1.30	1.52	53.9
LDA	1.02	0.37	0.65	63.4
NB	0.30	0.13	0.18	58.6
k-NN	8.74	1.96	6.78	77.6
RF	119.94	67.18	52.76	44.0
SVC	663.56	431.57	231.99	35.0
ABC	28.98	21.07	7.91	27.3
DT	3.44	2.59	0.85	24.8
EE	297.52	210.91	86.61	29.1
RUSB	39.08	26.81	12.27	31.4
BRF	106.98	77.15	29.83	27.9
BB	50.64	37.55	13.09	25.9
HGB	9.62	4.69	4.92	51.2
ANN (@ 10 epochs)	237.47	12.25	225.22	94.8
DNN (@ 10 epochs)	320.02	12.12	307.90	96.2
Optimized DNN (@ 10 epochs)	250.98	13.36	237.62	94.7

distances and the number of neurons in each layer in the neural network.

In general, training neural networks is time and computational resource consuming. The training time of the neural network is training time per epoch multiplied by the number of epochs required to achieve the optimal solution [66]. An epoch is the single forward and backpropagation pass through all the training dataset, usually in predefined batch size [66].

To access the impact computational environment has on-time complexity in terms of the hardware configuration, we tested on two different hardware configurations: 1) Intel i3 CPU with 6GB RAM, and 2) Kaggle's cloud platform with 4 CPU core and 16GB RAM, similar to the works in [66], [67]. The time complexity gains as shown in Table 9 proves that apart from the dataset size and machine learning

algorithm, the computational environment does have a significant impact on the runtime of an algorithm, which is in agreement with earlier studies [66], [67]. It is also worth noting that applying preprocessing strategies (MV, resampling imbalanced data and data normalization) before training the models do also have a positive impact on speeding up the computational time.

We observe in Table 9 that even though methods such as linear regression and GaussianNB, had lower time complexity in comparison to the other models and the proposed DNN, their prediction power suffered in the process because they are not able to take account of the exact and intrinsic properties of the dataset in comparison to the other more complex models. This is evident based on the results obtained in Tables 3-4, as well as with the analysis of the statistical test.

V. DISCUSSION

In this section, we summarize our findings and draw some conclusions given the experiments conducted as follows:

1) Applying missing value benefited all the classifiers, as some classifiers such as k-NN and SVM do not work in the presence of missing values, unlike the tree-based methods that can handle missing values. Hence the better performance of the tree-based methods, especially RF, even with an imbalance training set. The statistical test also supports this observation.
2) Upon applying both missing values and resampling methods, the neural network methods were the better performers, except for methods like LR and LDA because they are less able to capture the intrinsic characteristics of the dataset. However, the neural network methods are the most sensitive to imbalance class distribution and showed the most bias toward the majority class.
3) An interesting observation was the slightly improved performance of the optimized DNN method over the ensemble methods, with the caveat that optimization was not performed on the ensemble methods, which could have given different results. The result seems to support the intuition that DNN implements functions of higher complexity, so that they are able, with the same number of resources, address more difficult problems [66]. However, for neural networks to generalize well, aside from training on a large amount of data, the test data must be similar to the training data, allowing the output decision to interpolate between the training set. This assumption is evident in the performance of the optimized DNN techniques when tested on an unseen imbalanced dataset different from the balanced data used during training.
4) Training a neural network includes both forward and backward propagation, whereas deploying a trained network on a test set (unseen new data points) involves only forward propagation. Thus, estimating the execution time of model training incorporates both model training and deployment [67]. Hence, we can infer that predicting the execution time for the entire deep neural network is possible by combining these two execution time results. We can also infer that a low time complexity does not necessarily mean a high prediction power.
5) We also observed that rbf kernel SVM had the highest time computational complexity in comparison to the other models. This observation is also in-line with conclusions in the literature about the computational cost of the SVM algorithm. SVM is an effective learning algorithm, but its computation and storage requirement increases rapidly with the number of training vectors, based on its learning philosophy of separating support vectors from the rest of the training dataset. Hence, the complexity of SVM is assumed to be $O(n^3)$ and depends on the number of the training set, the number of features, type of kernel function and the regularization parameter [68], [69].
6) The experimental analysis was mainly performed on a binary classification problem. However, various methods exist to handle multi-class and multi-label as multiple binary classification tasks.

A. LIMITATION OF STUDY

In this study, we compared different strategies for mitigating the effect of imbalanced data with missing values and proposed an optimized DNN model as a way of achieving a better performance.

One apparent limitation of the proposed DNN method observed in this study is the high sensitivity of neural networks to an imbalanced dataset, in comparison to the tree-based k-NN algorithms. We observed the model showing bias to the majority class, such inherited biases in the training phase are also transferred into the test or prediction stage. Another possible limitation of DNN methods is what is often referred to as the 'black box problem', centering on transparency in how it arrived at a decision. This is very critical to know about problems like water quality anomaly detection.

The process of training neural networks is the most challenging aspect of applying the method, and in general is by far the most time consuming, both in terms of effort required to configure the process and computational complexity required to execute the process. Other limitations and challenging factors associated with of DNN method are well articulated in [70].

In this study, default settings were maintained during the entire experiments for all the other learning models. However, their performance could still be improved by conducting detailed parameters tuning, especially with the tree-based and ensemble models. Notwithstanding, the results obtained would serve as baseline performance for future improvements on the learning models for this particular dataset problem.

B. THREATS TO VALIDITY

In machine learning and data science experiments, a discussion on how threats to validity were dealt with is crucial [71].

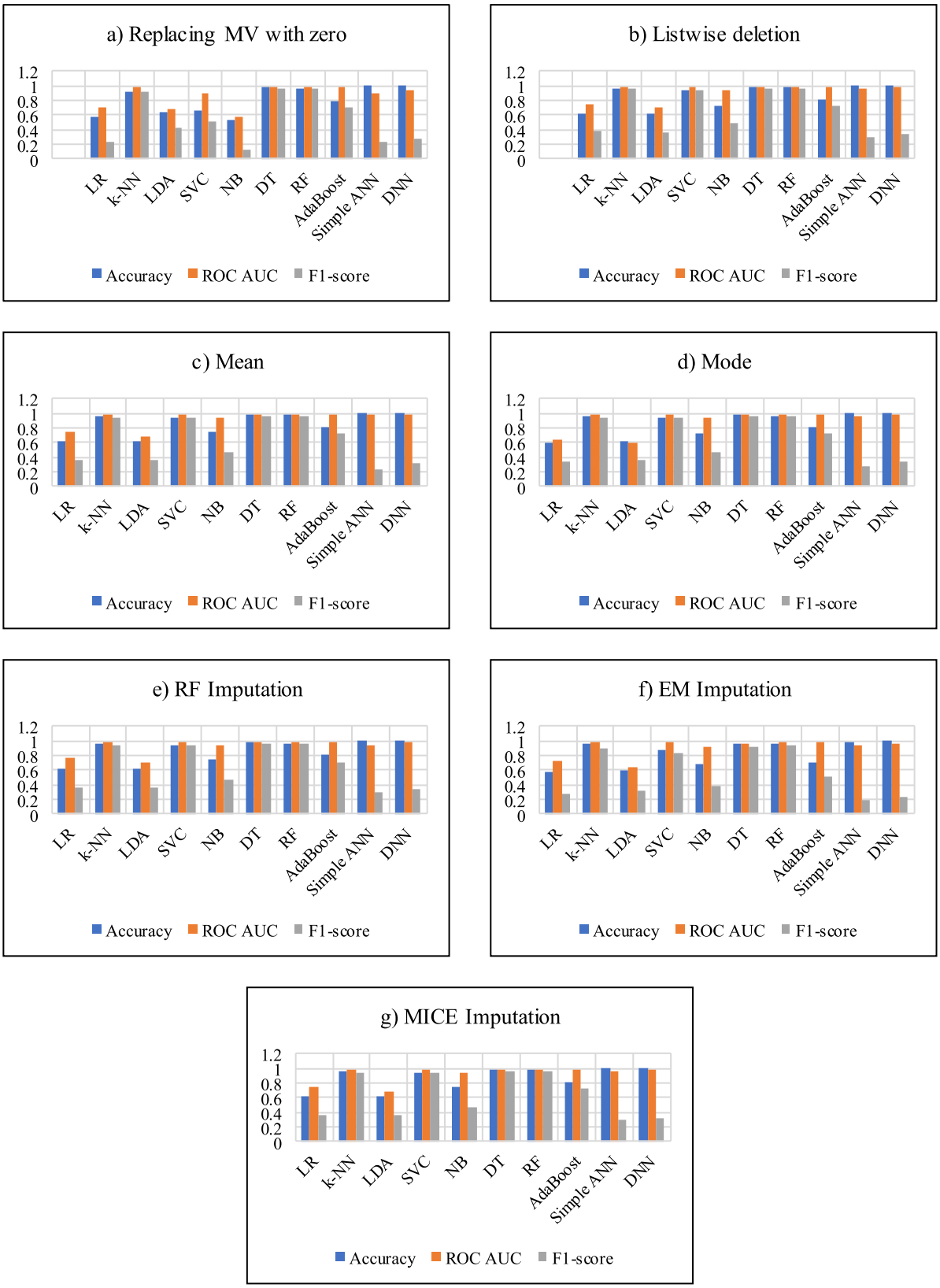


FIGURE 12. The graphical results of the training set applying only MV methods.

TABLE 10. Average rank of classifiers in terms of F1-measure.

LR	k-NN	LDA	SVC	NB	DT	RF	ABC	ANN	DNN
Ranks with missing values but no resampling methods									
7.86	3.00	7.00	4.14	6.57	2.00	**1.00**	4.88	9.86	8.71
Ranks with missing values and resampling methods									
9.00	6.00	8.625	4.5	6.375	7.125	5.125	5.125	2.00	**1.125**

This is to ensure that the dataset supports the conclusion being made and it is as a result of the effect of the treatment applied (MV, resampling and learning algorithm), and not just by chance.

A threat to internal validity is accounting for the influences that have an impact on the empirical results obtained. To mitigate threats to internal validity, a standard SPYDER Python data mining tool was used for all the algorithms experimented on, using Intel Xeon CPU@3.20GHz, 16GB RAM system. Secondly, all the parameters settings are outlined in Table 1 to allow for the reproducibility of the experiments. Before building all the learning algorithms, the dataset was first standardized, considering our dataset is composed of numeric features. This is important to ensure all the features lie within the same range, to avoid large-value features to have a dominating influence on the small-value ones. The parameter tuning for optimizing the DNN algorithm were selected using a grid search approach to achieve better performance on our dataset. To ensure confidence in the reliability of the experiments conducted, all the results obtained were also validated by all co-authors for accuracy.

Threats to external validity account for the generalization of the results obtained outside the experimental setting or framework, and the limits needed to be applied. In all our experiments, we utilized a real-world dataset from a reliable source, which in no small measure boosts the reliability of our conclusions. Analysing the dataset on several learning algorithms in addition to repeated cross-validation gives us confidence in the reliability of our experimental results and conclusions.

VI. CONCLUSION AND FUTURE WORK

In dealing with missing value and class imbalanced in water quality anomaly detection classification problem, this paper experimentally evaluated the empirical evidence based on the argument that a combination of missing value and resampling methods can improve the performance of classifiers due to the benefit they derive by implementing these preprocessing methods on the training set before fitting a learning model. We experimentally evaluated the performance of several homogenous and heterogeneous based static ensemble classifiers with the application of MV and ICD methods on the training set as well as optimized DNN model using grid search.

This paper aimed to observe the effects of combining these variations of strategies on the performance of classifiers specifically on two-class anomaly detection for drinking-water quality dataset problem. For the experiments, we considered seven missing data and eight resampling methods on ten different classifiers. The experimental results obtained revealed that classifiers benefit from combining MV and ICD preprocessing methods to enhance their classification performance in terms of the F1-measure metric. In particular, the tree-based RF algorithm algorithms consistently performed across a combination of these varying MV and ICD strategies. Generally, the performance of the entire ensemble models implementations was low when tested on the imbalanced test set. However, there was a slight improvement using DNN. This is because evaluating classifiers when applying MV and ICD methods on only the training is one case, whereas evaluating the classifiers on an imbalance (unseen) test set is a different case altogether that test the robustness of a given classifier. These experimental observations lead us to accept the two alternative hypotheses that missing values and imbalanced data harm performance metrics of learning classifiers and reject the null hypotheses.

TABLE 11. Average rank of the ensemble and optimized DNN methods considering all the three performance scores (balanced accuracy, ROC and F1 scores).

Method	Rank	Method	Rank
Ensemble 1 (ENSEM1)	13.83	Ensemble 11 (ENSEM11)	14.33
Ensemble 2 (ENSEM2)	13.17	Ensemble 12 (ENSEM12)	13.00
Ensemble 3 (ENSEM3)	11.00	Easy Ensemble (EE)	5.67
Ensemble 4 (ENSEM4)	9.17	RUSBoost (RUSB)	15.67
Ensemble 5 (ENSEM5)	12.00	Balance Random Forest (BRF)	10.00
Ensemble 6 (ENSEM6)	7.67	Balanced Bagging (BB)	7.67
Ensemble 7 (ENSEM7)	11.67	HistGradientBoosting (HGB)	7.67
Ensemble 8 (ENSEM8)	16.00	Stacking (STK1)	4.33
Ensemble 9 (ENSEM9)	19.67	Stacking (STK2)	7.50
Ensemble 10 (ENSEM10)	7.67	Optimized DNN	**2.33**

In future work, we aim to implement two dynamic selection techniques namely, dynamic classifier selection and dynamic ensemble selection methods [11], [72], that suits and improve the water quality anomaly detection prediction problem over the traditional voting and boosting approaches examined in this study. Additionally, DNN has shown promise based on the results obtained in this study. The authors would also like to pay detailed attention to DNN approaches [2], as an area of interest in future work.

APPENDIX A

Fig. 12 shows the graphical results when applying missing value methods on the training set to evaluate the classifiers.

APPENDIX B

The average rank of classifiers for the two analysed scenarios: interactions between classifiers, missing value and F1-measure; and interactions between classifiers, missing

values, resampling and F1-measure are shown in Table 10. The average rankings were used to show the CD diagram of the *post hoc* Nemenyi test results in Figs. 6-9.

The global average rank of ensemble and the proposed DNN models across the three performance scores (balanced accuracy, ROC and F1-score) are presented in Table 11. These average rankings were used to show the CD diagrams of the *post hoc* Nemenyi test depicted in Fig. 10 and Fig. 11.

ACKNOWLEDGMENT

The authors would like to thank the University of Johannesburg and the Durban University of Technology for making the resources available to complete this work. They are also thankful to Professor Thomas Bartz-Beielstein for providing the *Thüringer Fernwasserversorgung* water utility dataset used for conducting all the experiments.

REFERENCES

[1] F. Rehbach, S. Moritz, S. Chandrasekaran, M. Rebolledo, M. Friese, and T. Bartz-Beielstein. (2018). *GECCO 2018 Industrial Challenge: Monitoring of Drinking-Water Quality*. Accessed: Feb. 19, 2019. [Online]. Available: http://www.spotseven.de/wp-content/uploads/2018/03/rulesGeccoIc2018.pdf

[2] E. M. Dogo, N. I. Nwulu, B. Twala, and C. Aigbavboa, "A survey of machine learning methods applied to anomaly detection on drinking-water quality data," *Urban Water J.*, vol. 16, no. 3, pp. 235–248, Mar. 2019, doi: 10.1080/1573062X.2019.1637002.

[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, doi: 10.1109/TKDE.2008.239.

[4] J. L. Schafer and J. W. Graham, "Missing data: Our view of the state of the art," *Psychol. Methods*, vol. 7, no. 2, pp. 147–177, 2002, doi: 10.1037//1082-989X.7.2.147.

[5] I. F. Ilyas and C. Xu, *Data Cleaning*. San Rafael, CA, USA: Morgan & Claypool, 2019.

[6] G. Lemaitre, F. Nogueira, and C. Aridas, "Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 559–563, 2017.

[7] P. J. García-Laencina, J.-L. Sancho-Gómez, and A. R. Figueiras-Vidal, "Pattern classification with missing data: A review," *Neural Comput. Appl.*, vol. 19, no. 2, pp. 263–282, Mar. 2010, doi: 10.1007/s00521-009-0295-6.

[8] R. C. Prati, G. E. A. P. A. Batista, and D. F. Silva, "Class imbalance revisited: A new experimental setup to assess the performance of treatment methods," *Knowl. Inf. Syst.*, vol. 45, no. 1, pp. 247–270, Oct. 2015, doi: 10.1007/s10115-014-0794-3.

[9] S. S. Khan, A. Ahmad, and A. Mihailidis, "Bootstrapping and multiple imputation ensemble approaches for classification problems," *J. Intell. Fuzzy Syst.*, vol. 37, no. 6, pp. 7769–7783, Dec. 2019, doi: 10.3233/JIFS-182656.

[10] B. Krawczyk, "Learning from imbalanced data: Open challenges and future directions," *Prog. Artif. Intell.*, vol. 5, no. 4, pp. 221–232, Nov. 2016, doi: 10.1007/s13748-016-0094-0.

[11] A. Roy, R. M. O. Cruz, R. Sabourin, and G. D. C. Cavalcanti, "A study on combining dynamic selection and data preprocessing for imbalance learning," *Neurocomputing*, vol. 286, pp. 179–192, Apr. 2018, doi: 10.1016/j.neucom.2018.01.060.

[12] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing, "Learning from class-imbalanced data: Review of methods and applications," *Expert Syst. Appl.*, vol. 73, pp. 220–239, May 2017, doi: 10.1016/j.eswa.2016.12.035.

[13] P. Schmitt, J. Mandel, and M. Guedj, "A comparison of six methods for missing data imputation," *J. Biometrics Biostatist.*, vol. 6, no. 1, pp. 1–6, 2015, doi: 10.4172/2155-6180.1000224.

[14] T. Makaba and E. Dogo, "A comparison of strategies for missing values in data on machine learning classification algorithms," in *Proc. Int. Multidisciplinary Inf. Technol. Eng. Conf. (IMITEC)*, Vanderbijlpark, South Africa, Nov. 2019, pp. 1–7, doi: 10.1109/IMITEC45504.2019.9015889.

[15] M. Nguyen and D. Logofatu, "Applying tree ensemble to detect anomalies in real-world water composition dataset," in *Intelligent Data Engineering and Automated Learning* (Lecture Notes in Computer Science), vol. 11314, H. Yin, D. Camacho, P. Novais, and A. Tallón-Ballesteros, Eds. Cham, Switzerland: Springer, 2018, pp. 429–438.

[16] F. Muharemi, D. Logofatu, and F. Leon, "Machine learning approaches for anomaly detection of water quality on a real-world data set," *J. Inf. Telecommun.*, vol. 3, no. 9, pp. 294–307, 2019, doi: 10.1080/24751839.2019.1565653.

[17] V. H. A. Ribeiro and G. Reynoso-Meza, "Online anomaly detection for drinking water quality using a multi-objective machine learning approach," in *Proc. Genetic Evol. Comput. Conf. Companion*, Jul. 2018, pp. 1–2, doi: 10.1145/3205651.3208202.

[18] V. H. A. Ribeiro and G. Reynoso-Meza, "Monitoring of drinking-water quality by means of a multi-objective ensemble learning approach," in *Proc. Genetic Evol. Comput. Conf. Companion*, Jul. 2019, pp. 1–2.

[19] D. Wu, H. Wang, and R. Seidu, "Smart data driven quality prediction for urban water source management," *Future Gener. Comput. Syst.*, vol. 107, pp. 418–432, Jun. 2020, doi: 10.1016/j.future.2020.02.022.

[20] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," *IJCAI*, vol. 14, no. 2, 1995, pp. 1137–1145.

[21] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. 3rd ed. New York, NY, USA: Wiley, 2016.

[22] B. Twala and F. Mekuria, "Ensemble multisensor data using state-of-the-art classification methods," in *Proc. Africon*, Pointe-Aux-Piments, Mauritius, Sep. 2013, pp. 1–6.

[23] B. Twala, "An empirical comparison of techniques for handling incomplete data using decision trees," *Appl. Artif. Intell.*, vol. 23, no. 5, pp. 373–405, May 2009, doi: 10.1080/08839510902872223.

[24] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 4, pp. 463–484, Jul. 2012, doi: 10.1109/TSMCC.2011.2161285.

[25] W. Zhang, Y. Yang, and Q. Wang, "Handling missing data in software effort prediction with Naive Bayes and EM algorithm," in *Proc. 7th Int. Conf. Predictive Models Softw. Eng. Promise*, 2011, pp. 1–10.

[26] O. Mishchuk, R. Tkachenko, and I. Izonin, "Missing data imputation through SGTM neural-like structure for environmental monitoring tasks," in *Advances in Computer Science for Engineering and Education II* (Advances in Intelligent Systems and Computing), vol. 938, Z. Hu, S. Petoukhov, I. Dychka, and M. He, Eds. Cham, Switzerland: Springer, 2020, doi: 10.1007/978-3-030-16621-2_13.

[27] I. Izonin, N. Kryvinska, P. Vitynskyi, R. Tkachenko, and K. Zub, "GRNN approach towards missing data recovery between IoT systems," in *Proc. Adv. Intell. Netw. Collaborative Syst.*, L. Barolli, H. Nishino and H. Miwa, Eds. Cham, Switzerland: Springer, Cham, 2020, pp. 445–453.

[28] R. Tkachenko, I. Izonin, N. Kryvinska, I. Dronyuk, and K. Zub, "An approach towards increasing prediction accuracy for the recovery of missing IoT data based on the GRNN-SGTM ensemble," *Sensors*, vol. 20, no. 9, p. 2625, May 2020, doi: 10.3390/s20092625.

[29] C. Gautam and V. Ravi, "Data imputation via evolutionary computation, clustering and a neural network," (Amsterdam), *Neurocomputing*, vol. 156, pp. 134–142, May 2015, doi: 10.1016/j.neucom.2014.12.073.

[30] D. J. Stekhoven and P. Buhlmann, "MissForest-non-parametric missing value imputation for mixed-type data," *Bioinformatics*, vol. 28, no. 1, pp. 112–118, Jan. 2012, doi: 10.1093/bioinformatics/btr597.

[31] R. J. Little and D. B. Rubin, *Statistical Analysis With Missing Data*. Hoboken, NJ, USA: Wiley, 2019.

[32] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Roy. Stat. Soc., Ser. B (Methodol.)*, vol. 39, no. 1, pp. 1–38, 1977, doi: 10.1111/j.2517-6161.1977.tb01600.x.

[33] S. F. Buck, "A method of estimation of missing values in multivariate data suitable for use with an electronic computer," *J. Roy. Stat. Soc., Ser. B (Methodol.)*, vol. 22, no. 2, pp. 302–306, Jul. 1960, doi: 10.1111/j.2517-6161.1960.tb00375.x.

[34] I. Tomek, "Two modifications of CNN," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. SMC-6, no. 11, pp. 769–772, Nov. 1976, doi: 10.1109/TSMC.1976.4309452.

[35] I. Tomek, "An experiment with the edited nearest-neighbor rule," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. SMC-6, no. 6, pp. 448–452, Jun. 1976, doi: 10.1109/TSMC.1976.4309523.

[36] 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, doi: 10.1613/jair.953.

[37] 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. (IEEE World Congr. Comput. Intell.)*, Jun. 2008, pp. 1322–1328.

[38] G. E. Batista, A. L. Bazzan, and M. C. Monard, "Balancing training data for automated annotation of keywords: A case study," in *Proc. WOB*, 2003, pp. 10–18.

[39] 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 Explor. Newslett.*, vol. 6, no. 1, pp. 20–29, Jun. 2004, doi: 10.1145/1007730.1007735.

[40] J. I. E. Hoffman, "Logistic regression," in *Basic Biostatistics for Medical and Biomedical Practitioners*, 2nd ed. London, U.K.: Academic, 2019, ch. 33, pp. 581–589, doi: 10.1016/B978-0-12-817084-7.00033-4.

[41] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Mach. Learn.*, vol. 6, no. 1, pp. 37–66, Jan. 1991, doi: 10.1007/BF00153759.

[42] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop Comput. Learn. Theory (COLT)*, 1992, pp. 144–152.

[43] L. I. Kuncheva, "On the optimality of Naïve Bayes with dependent binary features," *Pattern Recognit. Lett.*, vol. 27, no. 7, pp. 830–837, May 2006, doi: 10.1016/j.patrec.2005.12.001.

[44] H. Zhang, "Exploring conditions for the optimality of Naïve Bayes," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 19, no. 2, pp. 183–198, Mar. 2005, doi: 10.1142/S0218001405003983.

[45] J. R. Quinlan, "Simplifying decision trees," *Int. J. Man-Mach. Stud.*, vol. 27, no. 3, pp. 221–234, Sep. 1987, doi: 10.1016/S0020-7373(87)80053-6.

[46] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[47] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. Int. Conf. Mach. Learn.*, vol. 96. 1996, pp. 148–156.

[48] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[49] B. Krawczyk and G. Schaefer, "An improved ensemble approach for imbalanced classification problems," in *Proc. IEEE 8th Int. Symp. Appl. Comput. Intell. Informat. (SACI)*, May 2013, pp. 423–426.

[50] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 39, no. 2, pp. 539–550, Apr. 2009, doi: 10.1109/TSMCB.2008.2007853.

[51] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 40, no. 1, pp. 185–197, Jan. 2010, doi: 10.1109/TSMCA.2009.2029559.

[52] C. Chao, A. Liaw, and L. Breiman. (2004). *Using Random Forest to Learn Imbalanced Data*. Accessed: Aug. 15, 2020. [Online]. Available: https://statistics.berkeley.edu/sites/default/files/tech-reports/666.pdf

[53] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, Aug. 1996, doi: 10.1007/BF00058655.

[54] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, Aug. 1998, doi: 10.1109/34.709601.

[55] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T. Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3146–3154.

[56] S. Raschka, "MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack," *J. Open Source Softw.*, vol. 3, no. 24, p. 638, Apr. 2018, doi: 10.21105/joss.00638.

[57] D. H. Wolpert, "Stacked generalization," *Neural Netw.*, vol. 5, no. 2, pp. 241–259, 1992, doi: 10.1016/S0893-6080(05)80023-1.

[58] P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," *ACM Comput. Surv.*, vol. 49, no. 2, pp. 1–50, Nov. 2016, doi: 10.1145/2907070.

[59] C. Ferri, J. Hernández-Orallo, and R. Modroiu, "An experimental comparison of performance measures for classification," *Pattern Recognit. Lett.*, vol. 30, no. 1, pp. 27–38, Jan. 2009, doi: 10.1016/j.patrec.2008.08.010.

[60] Y. Liu, Y. Zhou, S. Wen, and C. Tang, "A strategy on selecting performance metrics for classifier evaluation," *Int. J. Mobile Comput. Multimedia Commun.*, vol. 6, no. 4, pp. 20–35, Oct. 2014, doi: 10.4018/IJMCMC.2014100102.

[61] M. Denil and T. Trappenberg, "Overlap versus imbalance," in *Proc. Can. Conf. Artif. Intell.* Berlin, Germany: Springer, 2010, pp. 220–231.

[62] A. Narassiguin, M. Bibimoune, H. Elghazel, and A. Aussem, "An extensive empirical comparison of ensemble learning methods for binary classification," *Pattern Anal. Appl.*, vol. 19, no. 4, pp. 1093–1128, Nov. 2016, doi: 10.1007/s10044-016-0553-z.

[63] T. Köse, S. Özgür, E. Cosgun, A. Keskinoglu, and P. Keskinoglu, "Effect of missing data imputation on deep learning prediction performance for vesicoureteral reflux and recurrent urinary tract infection clinical study," *BioMed. Res. Int.*, vol. 2020, pp. 1–15, Jul. 2020, doi: 10.1155/2020/1895076.

[64] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[65] S. Arora and B. Barak, *Computational Complexity: A Modern Approach*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[66] D. Justus, J. Brennan, S. Bonner, and A. S. McGough, "Predicting the computational cost of deep learning models," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 3873–3882.

[67] H. Qi, E. R. Sparks, and A. Talwalkar, *Paleo: A Performance Model for Deep Neural Networks*. Accessed: Oct. 10, 2020. [Online]. Available: https://openreview.net/pdf?id=SyVVJ85lg

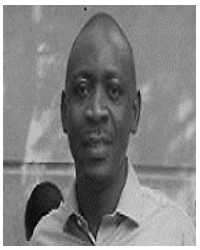
[68] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, "Fast kernel classifiers with online and active learning," *J. Mach. Learn. Res.*, vol. 6, pp. 1579–1619, Oct. 2005.

[69] A. Abdiansah and R. Wardoyo, "Time complexity analysis of support vector machines (SVM) in LibSVM," *Int. J. Comput. Appl.*, vol. 128, no. 3, pp. 28–34, Oct. 2015, doi: 10.5120/ijca2015906480.

[70] G. Marcus, "Deep learning: A critical appraisal," 2018, *arXiv:1801.00631*. [Online]. Available: http://arxiv.org/abs/1801.00631

[71] T. M. Khoshgoftaar, M. Golawala, and J. V. Hulse, "An empirical study of learning from imbalanced data using random forest," in *Proc. 19th IEEE Int. Conf. Tools Artif. Intell. (ICTAI)*, Oct. 2007, pp. 310–317.

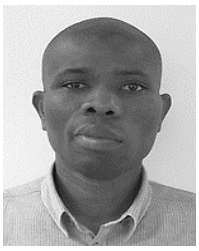
[72] R. M. O. Cruz, L. G. Hafemann, R. Sabourin, and G. D. C. Cavalcanti, "DESlib: A dynamic ensemble selection library in Python," 2018, *arXiv:1802.04967*. [Online]. Available: http://arxiv.org/abs/1802.04967



EUSTACE M. DOGO received the B.Sc. and M.Eng. degrees in electrical engineering from Peter the Great St. Petersburg Polytechnic University, Russia. He is currently pursuing the Ph.D. degree with the Department of Electrical and Electronic Engineering Science, Institute for Intelligent Systems, University of Johannesburg, South Africa. He is also Lecturer with the Department of Computer Engineering, Federal University of Technology Minna, Nigeria. His research interest includes theoretical and applied machine learning, currently focusing on water quality anomaly detection research.

NNAMDI I. NWULU received the B.Sc. and M.Sc. degrees in electrical and electronic engineering and the Ph.D. degree in electrical engineering. He is currently an Associate Professor with the Department of Electrical and Electronic Engineering Science, University of Johannesburg. His recent research interests include the application of mathematical optimization techniques and machine learning algorithms in the food, energy, and water spheres. He is also a registered Professional Engineer at the Engineering Council of South Africa (ECSA), a Senior Research Associate at the SARChI Chair in Innovation Studies, Tshwane University of Science and Technology, and an Associate Editor of the *African Journal of Science, Technology, Innovation and Development* (AJSTID).

BHEKISIPHO TWALA (Senior Member, IEEE) received the M.Sc. degree in statistics from the University of Southampton, U.K., in 2005, and the Ph.D. degree in machine learning and statistical science from Open University, U.K. He held a postdoctoral research position with Brunel University, U.K., mainly focusing on empirical software engineering research and further looking at data quality issues in software engineering. He is currently a Professor in Artificial Intelligence and Data Science and the Executive Dean of the Faculty of Engineering and Built Environment, Durban University of Technology (DUT), South Africa. He has held several prestigious positions, namely as the Director of the School of Engineering, University of South Africa, the Director of the Institute for Intelligent Systems, University of Johannesburg, the Head of the Department of Electrical and Electronic Engineering, University of Johannesburg. He is the author or coauthor of more than 180 journal articles, a book, book chapters, and other publications. His broad research interests include image and signal processing, multivariate statistics, applied and theoretical machine learning, knowledge discovery and reasoning with uncertainty, and the interface between statistics and computing. His many honors include the Prestigious Annual TW Kambule NSTF Research Award, in 2016, for his work in building on diverse experience in using artificial intelligence to solve several problems in many industries. He has made more than 50 keynote and related presentations at national and international forums. He is also the Editor-in-Chief and an Associate Editor of several prestigious international journals within his area of research.



CLINTON OHIS AIGBAVBOA received the Ph.D. degree in engineering management. He is currently a Full Professor of Sustainable Human Development with the Department of Construction Management and Quantity Surveying and the Director of the Sustainable Human Settlement and Construction Research Centre, Faculty of Engineering and the Built Environment, University of Johannesburg, South Africa. He has published over 500 research articles and several scholarly research books in his areas of interest. He has extensive knowledge in practice, research, training, and teaching. His recent research interest includes sustainable human settlement and construction research in the era of the fourth industrial revolution (4IR). He is an active postgraduate degree supervisor and has supervised over 40 masters and six Ph.D. students to completion. He is currently an Editor of the *Journal of Construction Project Management and Innovation* and has received national and international recognition in his field of research.