International Journal of Innovative Research and Scientific Studies, 6(3) 2023, pages: 545-552

IJIRSS
International Journal of Innovative Research & Scientific Studies

ISSN: 2617-6548

URL: www.ijirss.com

A fast and effective approach for classification medical data sets

Ivan Ivanov[1*], Borislava Toleva[2], Vincent James Hooper[3]

[1,2]*Faculty of Economics and Business Administration, Sofia University, Bulgaria.*
[3]*College of Business Administration, Prince Mohammad Bin Fahd University Al Khobar, Saudi Arabia.*

Corresponding author: Ivan Ivanov (*Email: i_ivanov@feb.uni-sofia.bg*)

Abstract

This study's objective is to offer a practical computer method for handling classification problems on large datasets. The aim of this study is to offer a practical computer approach for handling classification tasks on big datasets. We show that using Python's built-in parameters to balance classes can improve the accuracy and the metrics of a classification task. We employ logistic regression, support vector machines, decision trees, and random forest classifier. We use the parameter "class_weight='balanced'" to run each classification model as well as stratified train/test splitting to ensure that relative class frequencies are approximately preserved in each train and set subsets. We use our methods on medical datasets because class imbalance is frequently a problem there. Our research shows that the proposed algorithms can improve the accuracy and classification metrics of the given medical datasets. We propose an effective and easy-to-apply alternative to improve the prediction ability of the presented classification models in medical datasets. We test an easily reproducible set-up where any classification model can be used to model imbalanced classes. The key tuning of the model lies in the stratified train/test split as well as the parameter "class weight='balanced'". By combination of parameter tuning, better classification performance can be obtained in a quick and simple manner. It is simple and quick to replicate our algorithms to examine various medical datasets and determine which model best fits the data. It can be reproduced in biostatistical laboratories and by medical companies. Because it is simple to comprehend, medical researchers can swiftly review the information and determine the best course of action.

Keywords: Algorithms, Classification task, Confusion matrix, Machine learning, Models, Python, Diabetes Dataset.

DOI: 10.53894/ijirss.v6i3.1580
Funding: This research is supported by the College of Business Administration, Prince Mohammad Bin Fahd University Al Khobar, Saudi Arabia (Grant number: FRGVJH2023).
History: Received: 12 December 2022/**Revised:** 24 February 2023/**Accepted:** 19 April 2023/**Published:** 27 April 2023

Authors' Contributions: All authors contributed equally to the conception and design of the study.
Competing Interests: The authors declare that they have no competing interests.
Transparency: The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained.
Ethical Statement: This study followed all ethical practices during writing.
Publisher: Innovative Research Publishing

1. Introduction

In the past ten years, there has been an increase in research on big data, its nature, and applications. The definition of big data is ambiguous. Each definition reveals the goblins from a specific point of view. The diversity in the definition of big data shows its wide significance and numerous applications. Manyika et al explain big dataset as "whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze", i.e., the typical software_finds it difficult to analyze big datasets. The article [1] discusses different perspectives on big data and their applications in detail. We will think about the development of software capabilities to get around the challenges of big data analysis in the context of classification analysis, which is motivated by the definition of Manyika, et al. [2].

Python has become the most popular software in machine learning as it is relatively easy to learn and it has a big number of capacities. A bunch of Python libraries for machine learning (ML) exist along with deep learning libraries. Recently, the ML algorithms were combined into a new Python library called Data Science Toolkit (DST) [3]. The most commonly used ML libraries in Python are NumPy [4] pandas [5] scikit learn [6] TensorFlow [7] matplotlib [8]. Each of them contains attributes with methods or functions that enable the fast building and testing of the model. The Python Data Science Toolkit (DTS) [3] employs a number of ML classes that are based on the above libraries. If the usual ML applications are employed, such a bundled library enables the data scientist to use Python in a faster and more effective manner [3]. It has a number of classes related to data transformation and exploratory analysis, ML algorithms and data visualization [3]. This is the first Python library created that combines every tool a data scientist may ever need. Other programss like R and Matlab already provide similar libraries [3]. One approach to model improvement combines all required algorithms in a single software package.

Another way to improve ML algorithms is to create updated software packages and ML algorithms. For example, creating new Python libraries that enhance the ML models by modifying their equation [9]. The Python package called Py Kernel Logit [9] introduces the model Kernel Logistic Regression (KLR). The package was created to meet the need for ML models in the transport industry. It modified the logistic regression in a way that makes its usage more suitable for predicting transportation demand [9]. Therefore, machine learning algorithms can be modified and applied in software packages depending on the needs of the field. There are numerous examples of modified ML and deep learning algorithms that were created into independent Python modules for usage in medical datasets. [10, 11]. The open-source Python library EEGraph is applied by authors [10]. Another Python library is dicom_wsi [11]. It is used to convert a slide into DICOM format that has medical applications.

By adding procedures to address dataset problems like class imbalance, the ML algorithm can also be made to perform better. Medical datasets frequently face this problem, and several solutions have been put up. [12-14]. Solutions use a variety of techniques by giving each observation a different weight in order to balance the classes. [15, 16]. Other authors follow the non-modified versions of the ML algorithms but use cross validation to evaluate the performance of the classification model [17]. A number of algorithms developed to tackle imbalance data are also applied in medical datasets, for example the Synthetic Minority Over-sampling Technique SMOTE [18], the clustering-based under-sampling techniques (described as DBIG-US) [19] and Sample-Characteristic Oversampling Technique (SCOTE) [20]. You may find a more thorough summary of classification techniques for unbalanced data in [21, 22].

In this article we will consider the impact of big data on the development of software capabilities for their analysis. We will pay attention on classification analysis as the main task for data analysis. Building an analytical model of data that categorizes them and aiming to disclose the intrinsic links between the data constitutes classification analysis.

This process is performed by software program that commands the analytical models and methods for their solution. These computer instructions are applied to a portion of the large dataset called the training subset, where the model is trained and hidden connections between the features of the large dataset are found. We will implement our research through the open-source programming language Python 3.7 in the Anaconda distribution.

We will present a classical algorithm for conducting classification analysis of large datasets. In this algorithm, we will present the software enhancements offered by Python to increase the efficiency of classification analysis. It includes methodology that uses big data analysis – big data analytics, i.e. splitting the given data set into two subsets – train subset and test subset (sometimes it is called validation subset). The classical command for this is the train_test_split (parameters) command. We will use the most used models for classification analysis - Logistic Regression (LR), Support vector machine (SVM), Random Forest Classifier (RF), Decision Tree Classifier (DT).

It should be noted that works on classification analysis of massive data sets abound in the scientific literature, where new variations of well-known models like logistic regression, reference vectors, random forests, and decision trees are created. Deep learning models are also widely studied, most commonly neural network models. All of them increase the efficiency of the models and algorithms that implement them. This happens at the expense of complicating the model and the need for more calculations, which in turn involves more computer time. We will take into consideration the possibilities of enhancing the effectiveness of classification analysis by implementing new software advancements as an alternative to academic literature to develop efficient improvements on classical models.

Our algorithm ensembles the classic methodology for big data analysis and software commands to obtain effective splitting of the given dataset into two subsets. We'll demonstrate how to get a training subset so that, when the relevant model is trained on it, the test subset yields greater values for the recall parameter. In addition, we compare the results from the algorithm with other similar results from the literature.

2. Models and the Algorithm

2.1. Machine Learning Models

We consider the standard and well-known models to classify the big data sets. We apply the following models: logistic regression, support vector machine, random forest, and decision tree. Theoretical and practical implementation of these models can be found in many issues [23]. Here, we consider the computer application of these models for solving the classification task. We present two Python software commands that we use in our algorithm:

train_test_split(X, y, test_size=0.2, stratify=y).

The above command splits the independent variables (at X) and dependent variable y into two parts train:test as 80:20, i.e. 20% of observations belong to the test subset. In addition, the parameter "stratify" is applied to ensure that relative class frequencies is approximately preserved in each subset, i.e. train and test ones [24].

The next command explains how to apply a classification model. We use the parameter "class_weight='balanced'". More specially, we have:

LogisticRegression(solver='lbfgs', max_iter=5000, class_weight='balanced') for the logistic regression;

SVC(C=1,kernel='rbf', gamma='auto' ,class_weight='balanced') for the support vector machine with the kernel='rbf';

RandomForestClassifier (class_weight='balanced') for the Random Forest Classifier; and

DecisionTreeClassifier (class_weight='balanced') for the Decision Tree Classifier.

The parameter class_weight associates weights with classes. If the values are not given, all classes are assigned weight one. Moreover, the "balanced" model uses the values of y to automatically computed weights. More explanations can be found at [25].

To analyze the proposed algorithm for a classification model we have employed accuracy, precision, and recall. The confusion matrix is usually applied to evaluate a model. The elements of a confusion matrix are true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Their meaning and definitions can be found in each book and paper of the machine learning analysis, and for example [26, 27]. The positive observations are assigned with Class 1 and the negative ones with Class 2. These values are used to compute the accuracy, precision, recall metrics displayed in the confusion report [28]. The formulas are the following:

Accuracy = (TP+TN)/(TP+TN+FP+FN);
Precision (Class 1) = TP/ (TP+FP) = Precision [27, 29];
Precision (Class 2) = TN/ (TN+FN);
Recall (Class 1) = TP / (TP+FN) = Sensitivity [26, 29] = Recall [27];
Recall (Class 2) = TN / (TN+FP) = Specificity [26, 29].

The parameter Precision (Class 1) is defined as Precision in [27]. The parameter Recall (Class 1) is defined as Sensitivity [26] and Recall [27]. The parameter Recall (Class 2) is defined as Specificity [26]. The Sensitivity parameter provides information on the accuracy of only positive predictions. Specificity represents the accuracy only of negative predictions [26].

2.2. The Algorithm

The algorithm is described as follows:

1.Load the dataset.
2.Shuffle dataset in-place.
3. Define X as independent variables, y is the dependent variable.
4.Split the dataset with.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,stratify=y)
5.Define the model LR, SVM, RF or DT applying the definitions above.
6.Fit the model on the train subset.
7.Evaluate the model on the test subset computing the confusion matrix and the classification report.
8.End computations.

3.The Research Findings

We use public datasets for reproducibility purposes. We carry out experiments with two public data sets: breast cancer Wisconsin available in Wisconsin Breast Cancer Dataset [30] and diabetes dataset Frankfurt available in Frankfurt Diabetes Dataset [31]. All experiments are executed with Python 3.7 in Anaconda on a Laptop with 1.50 GHz Intel(R) Core (TM)and 8 GB RAM, running on Windows 10. In the experiments, we split the data into 80% data training and 20% data testing.

3.1. Example 1. Diabetes Dataset Frankfurt

The classification analysis of diabetes dataset from the hospital Frankfurt is provided in Abdulkader and Alazzawi [29]. We apply our algorithm to solve the classification task. We compare the computational results in Table 1. We apply our algorithm with the Logistic Regression (LR), Support Vector Machine (SVM), Random Forest (RF), and Decision Tree (DT). We compare our results to similar models which are obtained in Abdulkader and Alazzawi [29] (see Table 1. Looking at Table 1 we conclude that our algorithm achieves better results from the corresponding values obtained by Abdulkader and Alazzawi [29]. Moreover, our algorithm predicts two classes of the dataset better (see values of Recall (Class 1) and Recall (Class 2) for the Random Forest and Decision Tree).

Table 1.
Results for diabetes dataset Frankfurt.

Models	Our algorithm class_weight = 'balanced'	Results Table 5 [29]
LR		
Accuracy	0.77	0.775
Precision (Class 1)	0.85	0.57
Precision (Class 2)	0.64	
Recall (Class 1)	0.79	0.889
Recall (Class 2)	0.73	0.75
SVC with kernel='rbf'		
Accuracy	0.96	0.81
Precision (Class 1)	0.94	0.482
Precision (Class 2)	1.00	
Recall (Class 1)	1.00	0.992
Recall (Class 2)	0.88	0.972
Random forest		
Accuracy	0.98	0.99
Precision (Class 1)	0.98	0.605
Precision (Class 2)	0.99	
Recall (Class 1)	0.99	0.873
Recall (Class 2)	0.96	0.735
Decision tree		
Accuracy	0.97	
Precision (Class 1)	0.98	No results
Precision (Class 2)	0.95	
Recall (Class 1)	0.97	
Recall (Class 2)	0.97	

The results given in Abdulkader and Alazzawi [29] are based on the non-modified versions of the logistic regression, support vector machines and random forests. Similar to our research, the authors [29] split the dataset set into 80% training set and 20% test set. However, they do not use stratified cross validation. Therefore, they cannot ensure that relative class frequencies are approximately preserved in each train and set subsets as it was in the original dataset. We use the stratified cross validation as it guarantees that the training and test sets will have similar structure to that of the original dataset. In the original dataset, class imbalance is present, so we fix the parameter 'class_weight=balance' to make sure the classes are balanced when fitting the models. Class imbalance is not handled in Abdulkader and Alazzawi [29]. This can be seen in Table 1 where the precision and recall metrics in Abdulkader and Alazzawi [29] are higher for class 1, which is the prevailing class in the dataset.

Similar to Abdulkader and Alazzawi [29], we achieve accuracy of 0.775% using the LR and 0.99 using the RF. However, our version of the logistic regression and the random forest results in higher precision and recall metrics for the two classes than those in Abdulkader and Alazzawi [29]. For instance, our LR achieves precision of 0.85 for class 1 compared to 0.57 in Abdulkader and Alazzawi [29]. Class 2 has a precision score of 0.64 in our LR model, while [29] do not calculate the precision score for class 2. The recall measure for class 2 is similar in our results and Abdulkader and Alazzawi [29] , while the recall for class 1 in Abdulkader and Alazzawi [29] is better than our model. However, the overall performance of our 'balanced' LR is better than in Abdulkader and Alazzawi [29].

Table 1 also shows that the random forest classifier predicts classes 1 and 2 more accurately than in Abdulkader and Alazzawi [29]. The suggested random forest classifier can predict whether a new observation belongs to class 1 or class 2 more accurately because all classification metrics are better than [29]. Although the authors [29] do not perform calculations with the decision tree classifier, Table 1 shows that it also achieves vey good accuracy, precision and recall measures for the two classes. Therefore, handling class imbalances is very important for model performance. With this example, we show that the inbuilt Python parameter 'class_weight' can be enough to handle class imbalances.Therefore, researchers have access to an easy and fast way to tackle class imbalances in medical datasets. To further explore this finding, we present another example.

3.2. Example 2. Breast Cancer Wisconsin

We also consider the classification problem for the set of Wisconsin breast cancer observations [30]. There are 30 numeric attributes of features in the dataset. Dataset has 569 instances that divided into two classes: benign and malignant. The class Benign consists of 212 instances, and malignant consists of 357 instances. The target is to classify the tumor as malignant or benign.This set has been intensively studied by several authors. We will compare the proposed algorithm and its results with the results from the following studies [27, 29, 32-34].

The focus of the article [27] is on the Artificial Neural Network (ANN) model.

The ANN model is applied as a procedure of feature selection. The results are: the Recall of 95% and the Precision of 60.32% (see Table 4) [27].

The XGBoost model has the best result obtained in the investigation from Prastyo, et al. [32]. The metrics of XGBoost are Recall 96,75%, Precision 97,28%, and Accuracy 97,19%. Comparing with the values of Table 2 we conclude the similar values of Random Forest and Decision Tree are competitive to the XGBoost model.

Table 2.
Results for the breast cancer Wisconsin (Positive = malignant = 0, Negative = benign = 1)

Models	**Our algorithm class_weight = 'balanced'**	**Results** [32].
LR		
Accuracy	0.98	
Precision (Class 1)	0.98	No results
Precision (Class 2)	0.98	
Recall (Class 1)	0.98	
Recall (Class 2)	0.98	
SVC with kernel='rbf'		
Accuracy	0.93	0.956
Precision (Class 1)	0.87	
Precision (Class 2)	1.00	0.984
Recall (Class 1)	1.00	
Recall (Class 2)	0.86	0.94
Random forest		
Accuracy	0.99	0.956
Precision (Class 1)	1.00	
Precision (Class 2)	0.98	0.984
Recall (Class 1)	0.98	
Recall (Class 2)	1.00	0.94
Decision tree		
Accuracy	0.99	
Precision (Class 1)	1.00	No results
Precision (Class 2)	0.98	
Recall (Class 1)	0.98	
Recall (Class 2)	1.00	

In this example our classification algorithms have a similar performance to those in Prastyo, et al. [32]. The authors also use 80/20 train/test split proportion but instead of stratified cross validation, they use one-fold cross validation to split into training and test set and then five-fold cross validation on the training and test sets from step 1. This is the novelty in their research.

They do not use a specified algorithm to handle class imbalances. Their algorithms perform well as seen in Table 2. Our methodology has a similar performance to that in Prastyo, et al. [32]. Our approach, however, is significantly easier to implement because it fixes the 'class_weight' option to 'balanced' and makes use of stratified cross validation, which is frequently advised in unbalanced datasets. Additionally, applying the built-in Python features is much quicker.

In addition, we compare values of the confusion matrix obtained by our algorithm, and values from Table 4 [33], Table 3 [32] and results from Adebiyi, et al. [34].

The elements of all confusion matrices on the test set with 114 observations (80%:20% for train:test) are given in Table 3 (our algorithm), Table 4, Table 5, and Table 6 in this paper. It is easy to make comparison between Table 3 and other ones.

Table 3.
The entries of the confusion matrix of breast cancer Wisconsin dataset.

Models	**Our algorithm**			
	TN	**FN**	**FP**	**TP**
LR	52	0	1	61
SVM, rbf	55	8	0	51
Random forest	56	0	0	58
Decision tree	53	1	0	60

Table 4.
The entries of the confusion matrix of breast cancer Wisconsin dataset based on Table 4 [33].

Models	Table 4 [33]			
	TN	FN	FP	TP
AdaBoost	81	7	0	26
k-NN	88	1	0	25
Decision tree	78	10	2	24
BSVM	87	1	5	21

Table 5.
The entries of the confusion matrix of breast cancer Wisconsin dataset based on Table 3 [32] Notations are given in Prastyo, et al. [32].

Models	Results from Table 3 [32]			
	TN	FN	FP	TP
GNB	43	4	4	63
k-NN	44	1	3	66
SVM	46	4	1	63
DF	46	4	1	63
AdaBoost	44	2	3	65
GB	45	2	2	65
XGBoost	46	1	1	66
MLP	43	3	4	64

Table 6.
The entries of the confusion matrix of breast cancer Wisconsin dataset based at Adebiyi, et al. [34].

Models	Results obtained in [34]			
	TN	FN	FP	TP
LDA using random forest	44	3	2	65
Without LDA using random forest	44	3	3	64
LDA using SVM	44	3	1	66
Without LDA using SVM	43	4	5	62

Note that FP+FN is the number of wrongly predicted observations from all confusion matrices described in Table 3, Table 4, Table 5, and Table 6. According to Table 5 the min(FP+FN)=2 (XGBoost, Table 5). The min(FP+FN)=1 for k-NN model obtained by Laghmati et al. as shown in Table 4. Moreover, the min(FP+FN)=0 for the Random Forest model obtained in this paper, and min(FP+FN)=1 for Decision Tree and Logistic Regression with the proposed algorithm (see Table 3).

4. Results

Our study shows that Python built-in packages can be successfully used to handle imbalanced classes in medical datasets. Our study has two important parts that are crucial to ensure model performance.

The first one is the type of cross validation. We use stratified k-fold cross validation rather than the widely used k-fold cross validation. When k-fold cross validation is not stratified, it would split into training and test set on a random base. Some of the training and test sets may have observations related to the two classes, while others may have observations related to a small number of one of the classes and much higher number of the other class. The k-fold cross validation in this case may result in misleading model evaluation as only the prevalent class will be predicted correctly [23]. This is the case in Table 1 [29] where the authors of the study do not consider the class imbalance when splitting into training and test set. The prevalent class 1 is the one correctly predicted in most cases. However, when we consider the class imbalance by using the stratified k-fold cross validation, then both classes are correctly predicted, and the model performance is not biased Table 1.

The importance of the correct train/test split strategy can be also viewed in Table 2. The authors of the comparative study used another train/test split strategy to handle class imbalance [32]. Their results are like ours in the Wisconsin breast cancer dataset Table 2. As it is shown in Table 2, we use another resampling strategy, stratified k-fold cross validation, that also proves to work well in medical datasets with imbalanced classes. Therefore, we show that the stratified k-fold cross validation can be a reliable alternative to the strategy applied in Abdulkader and Alazzawi [29] to handle classes imbalance.

The second important part of our study that ensures good model performance is the tuning of the parameter 'class_weight' that is part of every classification model in Python. We fixed to 'balanced' in order to tell Python that the classes in the two examples are not balanced and it should assign weights inversely proportionately to their frequencies to balance them. When the parameter 'class_weight' is None, the default value in Python, then both classes are given equal weight. But that is an issue when one of the classes dominates the other class. Giving equal weight would mean that both classes have a similar frequency and have relatively similar non-skewed distributions. That, however, is not the case with imbalanced classes. Class imbalance means that the distributions of the two classes have been skewed and the frequencies

are not similar. So, the parameter 'class_weight' =balanced tries to attribute different weights to the prevailing and rare classes so that it can implicitly compensate for the imbalance [23, 25].

The studies of the authors we mentioned before [32-34] do not tune this parameter. This is an issue in table one [32] where the authors do not handle the class imbalance. So, our models perform better. The authors of Laghmati, et al. [33]; Adebiyi, et al. [34] also did not tune this parameter but they used cross validation strategy to handle class imbalance, so they get similar results to ours. However, the results we obtained in Table 2 were better than those in Prastyo, et al. [32]. This may be the result of the 'class_weight' parameter.

These two important parts of our algorithm can be applied to any classification model. We have applied them to the support vector machines, logistic regression, decision tree classifier and the random forest classifier. They are built in Python, so their application is very easy and straightforward. Our approach can be utilized to resolve class imbalances in intricate classification models when a quick solution is needed, which has significant practical consequences. No preliminary code preparation is required as all necessary functions are built-in the scikit learn package in Python. So, no further complications of the code and structure of the classification model is required. The output of the model is easy to interpret, and the results are reliable. Therefore, our study can be applied by biostatistical companies prior to their laboratory experiments to examine combinations of factors and their relation to the outcome of a particular disease.

In the future we would extend this research to other medical datasets, and to further improve the efficiency of the classification process. The setting of the experiment may remain the same in order to further highlight its benefits.

As a conclusion, we obtain a new computer algorithm to provide a classification analysis upon applying the standard machine learning models. The technique has a straightforward structure and is based on several parameter selections in order to optimize computer model executions. The dataset's division is further crucial to the outcomes.

References

[1] H. K. Yaseen and A. M. Obaid, "Big data: Definition, architecture & applications," *JOIV: International Journal on Informatics Visualization*, vol. 4, no. 1, pp. 45-51, 2020. https://doi.org/10.30630/joiv.4.1.292

[2] J. Manyika, M. Chui, B. Brown, and J. Bughin, "Big data: The next frontier for innovation. Competition," Retrieved: https://www.semanticscholar.org/paper/Big-data%3A-The-next-frontier-for-innovation%2C-and-Manyika/91b63db746becca15090963a8990dfe2b5103799. 2011.

[3] C. El Hachimi, S. Belaqziz, S. Khabba, and A. Chehbouni, "Data Science Toolkit: An all-in-one python library to help researchers and practitioners in implementing data science-related algorithms with less effort," *Software Impacts*, vol. 12, p. 100240, 2022. https://doi.org/10.1016/j.simpa.2022.100240

[4] S. Van Der Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: A structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, no. 2, pp. 22-30, 2011. https://doi.org/10.1109/mcse.2011.37

[5] W. Mckinney, "Pandas: A foundational python library for data analysis and statistics," Retrieved: https://dlr.de/sc/portaldata/15/resources/dokumente/pyhpc2011/submissions/pyhpc2011_submission_9.pdf. [Accessed 2023.

[6] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *the Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.

[7] M. Abadi, "Tensorflow: A system for large-scale machine learning," Retrieved: https://research.google/pubs/pub45381/. 2016.

[8] P. Barrett, J. Hunter, J. Miller, J.-C. Hsu, and P. Greenfield, "matplotlib--A portable python plotting package," *Astronomical Data Analysis Software and Systems XIV*, vol. 347, no. 1, pp. 91-96, 2005.

[9] J. Á. Martín-Baos, R. García-Ródenas, and L. Rodriguez-Benitez, "Apython package for performing penalized maximum likelihood estimation of conditional logit models using kernel logistic regression," *Transportation Research Procedia*, vol. 58, pp. 61-68, 2021. https://doi.org/10.1016/j.trpro.2021.11.009

[10] A. M. Maitin, A. Nogales, P. Chazarra, and Á. J. García-Tejedor, "EEGraph: An open-source Python library for modeling electroencephalograms using graphs," *Neurocomputing*, vol. 519, pp. 127-134, 2023. https://doi.org/10.1016/j.neucom.2022.11.050

[11] Q. Gu, N. Prodduturi, J. Jiang, T. J. Flotte, and S. N. Hart, "Dicom_wsi: A python implementation for converting whole-slide images to digital imaging and Communications in Medicine compliant files," *Journal of Pathology Informatics*, vol. 12, no. 1, p. 21, 2021. https://doi.org/10.4103/jpi.jpi_88_20

[12] C.-L. Liu and P.-Y. Hsieh, "Model-based synthetic sampling for imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 8, pp. 1543-1556, 2019. https://doi.org/10.1109/tkde.2019.2905559

[13] N. Liu, X. Li, E. Qi, M. Xu, L. Li, and B. Gao, "A novel ensemble learning paradigm for medical diagnosis with imbalanced data," *IEEE Access*, vol. 8, pp. 171263-171280, 2020. https://doi.org/10.1109/access.2020.3014362

[14] X.-Y. Jing *et al.*, "Multiset feature learning for highly imbalanced data classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 139-156, 2019. https://doi.org/10.1609/aaai.v31i1.10739

[15] M. Zhu *et al.*, "Class weights random forest algorithm for processing class imbalanced medical data," *IEEE Access*, vol. 6, pp. 4641-4652, 2018. https://doi.org/10.1109/access.2018.2789428

[16] D. Gan, J. Shen, B. An, M. Xu, and N. Liu, "Integrating TANBN with cost sensitive classification algorithm for imbalanced data in medical diagnosis," *Computers & Industrial Engineering*, vol. 140, p. 106266, 2020. https://doi.org/10.1016/j.cie.2019.106266

[17] N. W. S. Wardhani, M. Y. Rochayani, A. Iriany, A. D. Sulistyono, and P. Lestantyo, "Crossvalidation metrics for evaluating classification performance on imbalanced data," in *2019 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, 2019, pp. 14–8.

[18] Z. Xu, D. Shen, T. Nie, and Y. Kou, "A hybrid sampling algorithm combining M-SMOTE and ENN based on random forest for medical imbalanced data," *Journal of Biomedical Informatics*, vol. 107, p. 103465, 2020. https://doi.org/10.1016/j.jbi.2020.103465

[19] A. Guzmán-Ponce, J. S. Sánchez, R. M. Valdovinos, and J. R. Marcial-Romero, "DBIG-US: A two-stage under-sampling algorithm to face the class imbalance problem," *Expert Systems with Applications*, vol. 168, p. 114301, 2021. https://doi.org/10.1016/j.eswa.2020.114301

[20] J. Wei, H. Huang, L. Yao, Y. Hu, Q. Fan, and D. Huang, "New imbalanced bearing fault diagnosis method based on Sample-characteristic Oversampling TechniquE (SCOTE) and multi-class LS-SVM," *Applied Soft Computing*, vol. 101, p. 107043, 2021. https://doi.org/10.1016/j.asoc.2020.107043

[21] V. S. Spelmen and R. Porkodi, "A review on handling imbalanced data," in *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT). IEEE*, 2018, pp. 1-11.

[22] D. Ramyachitra and P. Manikandan, "Imbalanced dataset classification and solutions: A review," *International Journal of Computing and Business Research*, vol. 5, no. 4, pp. 1-29, 2014.

[23] G. James, D. Witten, T. Hastie, and R. Tibshirani, *Introduction. In: An introduction to statistical learning. Springer Texts in Statistics*. New York: Springer, 2021.

[24] Cross Validation, "Python documentation. 3.1. Cross-Validation: Evaluating Estimator Performance — Scikit-Learn 1.2.0 Documentation," Retrieved: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html. [Accessed June 2022], 2022.

[25] Logistic Regression, "Python documentation," Retrieved: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html. 2022.

[26] D. Tripathi, D. R. Edla, V. Kuppili, A. Bablani, and R. Dharavath, "Credit scoring model based on weighted voting and cluster based feature selection," *Procedia Computer Science*, vol. 132, pp. 22-31, 2018.

[27] N. M. Lanbaran and E. Celik, "Prediction of breast cancer through tolerance-based intuitionistic fuzzy-rough Set Feature selection and artificial neural network," *Gazi University Journal of Science*, vol. 34, no. 4, pp. 1064-1075, 2021.

[28] Classification Report, "Python documentation," Retrieved: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html. [Accessed June 2022], 2022.

[29] R. M. Abdulkader and A. P. D. A. K. Alazzawi, "A comparison of five machine learning algorithms in the classification of diabetes dataset," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 14, pp. 1244-1259, 2021.

[30] Wisconsin Breast Cancer Dataset, "Breast cancer wisconsin," Retrieved: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29. 2022.

[31] Frankfurt Diabetes Dataset, "Diabetes," Retrieved: https://www.kaggle.com/code/emrzcn/diabetes-prediction-with-lr-knn-nb-svm-rf-gbm/notebook. [Accessed June 2022], 2022.

[32] P. H. Prastyo, I. G. Y. Paramartha, M. S. M. Pakpahan, and I. Ardiyanto, "Predicting breast cancer: A comparative analysis of machine learning algorithms," presented at the International Conference on Science and Engineering, 2020.

[33] S. Laghmati, B. Cherradi, A. Tmiri, O. Daanouni, and S. Hamida, "Classification of patients with breast cancer using neighbourhood component analysis and supervised machine learning techniques," presented at the 3rd International Conference on Advanced Communication Technologies and Networking (CommNet), 2020.

[34] M. O. Adebiyi, M. O. Arowolo, M. D. Mshelia, and O. O. Olugbara, "A Linear discriminant analysis and classification model for breast cancer diagnosis," *Applied Sciences*, vol. 12, no. 22, p. 11455, 2022. https://doi.org/10.3390/app122211455