A Low Resource Machine Learning Approach for Prediction of Dressler Syndrome

Diganta Sengupta, Subhash Mondal, Debosmita Chatterjee, Susmita Pradhan, and Pretha Sur

Abstract Cosmopolitan lifestyle and livelihood modifications have marked a toll on human health to the extent of myocardial disease onset at a relatively tender stage. One of the major issues that have been observed on the rise is the arterial blockage leading to myocardial infarction. Immune response to the arterial damage or the pericardium is termed as Dressler syndrome. This study focuses on prediction of Dressler syndrome based on myocardial infarction historical data. Moreover, the study focuses on prediction using a resource constraint dataset through six popular machine learning (ML) algorithms. The dataset comprised of 124 features, and 1700 data, post-cleaning. Of all the 124 features, 12 features were target values. We selected one of the target values (Dressler syndrome) for this study. 10% of the data was reserved for test data at the initial stage itself, and the rest was further split into 0.7:0.3 for training and validation sets. RF presented a model accuracy of 98%, which is the best of all the six algorithms. In terms of AUC, RF exhibited the highest value of 0.995. Moreover, the models were further tuned, and the results confirmed the efficacy of RF for the classification of Dressler syndrome.

D. Sengupta (✉) · S. Mondal · D. Chatterjee · S. Pradhan · P. Sur
Department of Computer Science and Engineering, Meghnad Saha Institute of Technology, Kolkata 700150, India
e-mail: sg.diganta@ieee.org

S. Mondal
e-mail: subhash@msit.edu.in

D. Chatterjee
e-mail: debosmita_c.cse2019@msit.edu.in

S. Pradhan
e-mail: susmita_p.cse2019@msit.edu.in

P. Sur
e-mail: pretha_s.cse2019@msit.edu.in

D. Sengupta
Department of Computer Science and Business Systems, Meghnad Saha Institute of Technology, Kolkata 700150, India

S. Bhattacharyya et al. (eds.), *Intelligent Systems and Human Machine Collaboration*, Lecture Notes in Electrical Engineering 985,
https://doi.org/10.1007/978-981-19-8477-8_6

Keywords Myocardial infarction · Dressler syndrome · Machine learning · Artificial intelligence · Bioinformatics

1 Introduction

Myocardial infarction popularly known as heart attack or cardiac arrest accounts for over one-fourth of the present annual global fatality [1]. Clinically it has been proven that the process initiates with a decline of blood inflow to the heart muscles. Multiple reasons have been cited till date for the decline such as arterial blockage due to cholesterol sedimentation in the arteries, excessive alcohol intake followed by poor diet, excessive stress, blood clotting, and in some cases cellular waste leading to the blood clot [2]. Present work stress followed by changing socio-economic lifestyle has aided in the growth of the decline parameters leading to the major share of fatalities through myocardial infarction. Post-myocardial infarction, the human immune system tries to initiate self-healing measures against the trauma caused to the heart muscles. This leads to inflammation of the membrane that encapsulates the heart (pericardium). This inflammation is clinically termed as pericarditis which is a common symptom of post-myocardial infarction. Another common symptom is the swelling of the pleurae leading to immense pain (pleuritic pain), and fever. All these symptoms taken together are clinically termed as Dressler syndrome. It has been observed that Dressler syndrome generally results from heart surgery, chest trauma, and myocardial infarction. Also it has been seen that the syndrome affects an age bracket of 20–50 years [3]. Also it has been observed that owing to a wide range of clinical presentations is usually tough for health professionals to recognize.

Dressler Syndrome being an immune system reaction may also lead to fluid build-up in the surrounding tissues of lungs also known as pleural effusion [4]. The build-up can put pressure on the heart muscles compelling them to work hard [4]. Chronic pathological inflammation can cause the pericardium to become scarred or thick, because of this heart's inability to efficiently pump blood [4]. So, it becomes important to timely identify Dressler Syndrome to minimize further risks for the patients. This served as the motivation for the study. We classify Dressler syndrome using Machine Learning (ML) algorithms based on historical data related to myocardial infarction leading to Dressler syndrome. Thereafter we claim that if Dressler syndrome is observed in a patient, then either the patient has had a myocardial infarction or is going to experience myocardial infarction.

We have chosen ML algorithms for this study because the dataset is resource constraint [5] which contains approximately a total of 1700 samples, the details of which are presented further in the paper. Multiple approaches have been applied to extract the best way of determining whether post-myocardial infarction, Dressler syndrome can occur or not. We present only the best results in this paper, excluding the other approaches we did which resulted in lower performance metrics results. Six ML algorithms have been used in this study as follows: Random Forest (RF), Xtreme Gradient Boost (XGB), Support Vector Machine (SVM), Decision Tree (DT), K Nearest Neighbor (KNN), and Logistic Regression (LR). The performance metrics which serve as the parameter of evaluation for the ML algorithms are accuracy, recall, precision, F1-score, and AUC score.

To the best of our knowledge, this is the first study which proposes a binary classifier model based on conventional ML algorithms which presents whether a patient has experienced or is going to experience myocardial infarction, based on a diagnosis of Dressler syndrome.

The rest of the paper is organized as follows. The next section presents the related work with respect to this study followed by the proposed classification models including the data preprocessing techniques presented in Sect. 3. The results are presented in Sect. 4 followed by the Discussion and Conclusion in Sect. 5.

2 Related Work

Although the study in this paper is novel, we present a few related works of importance in terms of myocardial infarction and Dressler syndrome. Authors in [1] have used ECG (electrocardiogram) signals for the prediction of myocardial infarction. The ECG signals have been decomposed in wavelets, thereby generating different clinical components within different sub-bands of the wavelets which are captured by Eigen space-based features, and wavelet entropy. In that study, KNN evolved as the best classifier seconded by SVM. They also presented a comparative analysis with convolutional neural networks. Their study focused on the prediction of myocardial infarction through wavelet decomposition of ECG signals using ML algorithms. The use of ECG signals for the prediction of myocardial infarction has been further proposed in [2, 6], using ML algorithms, in [7, 8] using DL algorithms. Authors in [8] also used Recurrent Neural Networks (RNN) for the prediction. Low-quality ECG signals have been used for the early detection of myocardial infarction in [9]. The authors have used DL frameworks for detection. Another approach for detection of myocardial infarction using DL algorithms is presented in [10] where the authors provide a two-fold approach, one class-based approach and another subject-based approach. Other ML-based approaches can be found in [11, 12].

Another approach for the prediction of myocardial infarction using ML algorithms is presented in [13]. The authors used a resource-constrained dataset containing a feature set of 26, and 345 instances. Three classes were presented in the dataset, namely Distinctive, Non-distinctive, and both (Distinctive and Non-Distinctive). Basically this study focused on multi-class prediction of myocardial infarction using ML algorithms such as Bagging, LR, and RF. The authors claimed accuracies of 93.91%, 93.63%, and 91.02%, respectively, for the three ML algorithms. Authors in [14] also predicted myocardial infarction using ECG signals, in which they generated a feature set containing twenty-one time domain features which had been extracted from ECG signals. This study focused on the use of Deep Learning (DL) algorithms such as Long Short-Term Memory (LSTM), and Convolutional Neural Networks (CNN). Their results exhibited training and testing accuracy of 99.05%, and 98.50%, respectively, using CNN and Bidirectional LSTM. Another noted work using LSTM can be found in [15]. Authors in [16] have done a comparative analysis of the detection of myocardial infarction using ML and DL algorithms. They used SVM in one study and Artificial Neural Networks (ANN) in another. They claim that SVM fared better than the DL counterpart.

3 Proposed Work

In this section we present the proposed workflow used for classification of Dressler Syndrome. Initially the data analysis comprised of three parts as discussed in Sect. 3.1 through Sect. 3.2. Then the model was trained using the ML algorithms. The workflow is presented in Fig. 1.

3.1 Dataset Acquisition

This present study was conducted using the dataset from [5] which comprised of 1700 instances and 124 features. Of the 124 features, the first 111 features ranging from column 2 to column 112 are input features for classification or prediction. The rest 12 columns from column 113 to column 124 contain target labels which denote complications that can arise from myocardial infarction. The dataset is a recent dataset and can be used for both classification as well as prediction. In this study only one of those 12 target labels has been used the Dressler syndrome. The dataset contains missing values which have been handled using data preprocessing as discussed in the subsequent subsections.

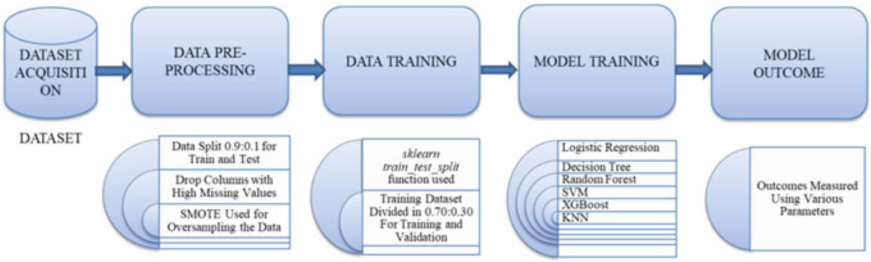

Fig. 1 Proposed workflow

3.2 Data Pre-processing

The dataset was split into two parts randomly to generate the test and the train set. As discussed earlier, a number of approaches had been used for splitting. We had used cross-validation to finalize the train test split ratio. Finally we focused on a split ratio of train to test as 0.9:0.1. Hence 10% of the dataset was used for testing, and 90% of the data was used for training as well as validation purpose. Column number 120 contained the Dressler syndrome. Hence barring column 120, we dropped the other 11 label columns from the study. Due to the high percentage of missing values of a particular column, we have also dropped one input column labeled IBS_NASL. Although other feature columns too contained missing values, but their count being admissible, we retained those features and handled them. Also it may be noted that the dataset is highly imbalanced containing 1462, and 68 values for the binary classes of 0, and 1, respectively. Hence, this class imbalance was also handled using the popular oversampling technique called SMOTE (Synthetic Minority Oversampling Technique). The application of SMOTE technique resulted in oversampled instances of 1462 values for each of the two classes, respectively.

3.3 Data/Model Training

For training the model, as discussed earlier, six ML algorithms were used. Initially the training dataset was split into a train and validate dataset using a ratio of 0.7:0.3, respectively. The training of the models was done using a popular ML library *sklearn* [17]. The ratio for 0.7:0.3 was again obtained using cross-validation in a random manner. The complete dataset comprised of 1700 instances. As 10% (170) of the instances were used as testing, the remaining 1530 instances were used for training and validation having the count of 1071 and 459, respectively. The choice of the six ML algorithms was based on prior art which contained prediction, and classification of myocardial infarction as discussed in the Related Work section. Out of the existing literature, the top six best performing algorithms in terms of the performance metrics were chosen for the study. The performance of the algorithms can be obtained from the related papers cited in the Related Work section.

4 Result Analysis

This section presents the results obtained from the ML algorithms in terms of their performance on the processed dataset. As discussed earlier, five performance metrics have been used to evaluate the performances. Table 1 presents the results thus obtained.

The results from Table 1 are graphically presented in Figs. 2, 3, 4, 5, and 6, respectively. It can be observed that although RF exhibits the best results in terms of all the performance metrics. Even the F1-Score for RF stands the best which is further established through the ROC-AUC score.

The ML models were further trained using the hyper-parameter tuned values. Tables 2 and 3 present the performance values with respect to the tuned versions. Table 2 presents the results with respect to *RandomisedSearchCV*, and Table 3 presents with respect to *GridSearchCV*. The tuned study was done to further validate the decision that RF generates the best classification result.

Table 1 Performance metric analysis for the respective machine learning models

Model	Accuracy (%)	Precision	Recall	F1-score	ROC-AUC score
LR	0.82	0.86	0.87	0.81	0.99
DT	0.92	0.94	0.94	0.92	0.993
RF	0.97	1	1	0.97	0.98
SVM	0.97	0.95	1	0.97	1
XGB	0.97	1	0.94	0.97	0.98
KNN	0.83	0.75	1	0.85	0.97

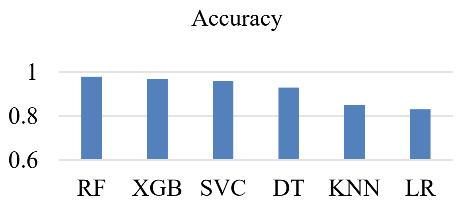

Fig. 2 Performance evaluation of six ML algorithms in terms of accuracy

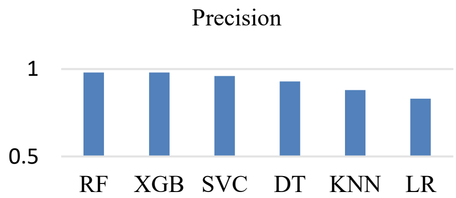

Fig. 3 Performance evaluation of six ML algorithms in terms of precision

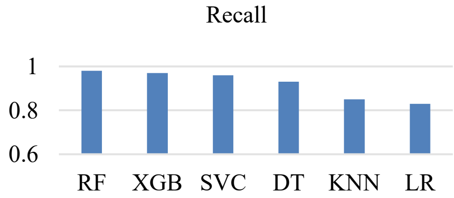

Fig. 4 Performance evaluation of six ML algorithms in terms of recall

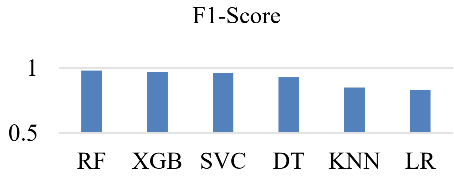

Fig. 5 Performance evaluation of six ML algorithms in terms of F1-score

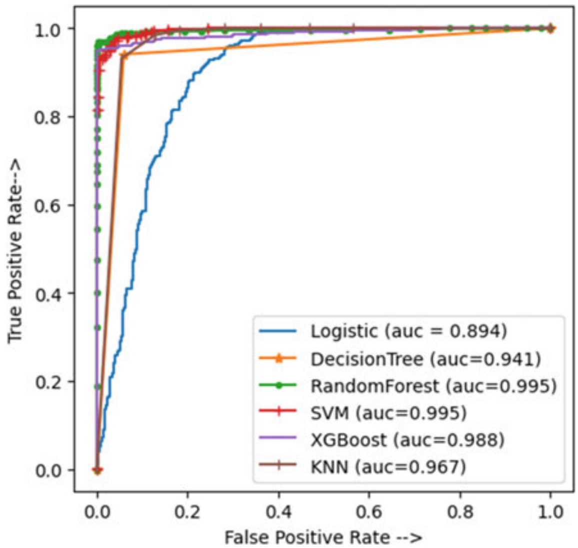

Fig. 6 Performance evaluation of six ML algorithms in terms of ROC-AOC score

Table 2 Performance metric analysis for the tuned models with *RandomisedSearchCV*

Model	Accuracy (%)	Precision	Recall	F1-score	ROC-AUC score
LR	0.84	0.86	0.9	0.83	0.9
DT	0.95	0.94	0.93	0.92	0.93
RF	0.98	1	0.95	0.975	0.96
SVM	0.85	0.81	0.93	0.86	0.85
XGB	0.97	0.996	0.95	0.97	0.97
KNN	0.86	0.79	1	0.85	0.96

Table 3 Performance metric analysis for the tuned models with *GridSearchCV*

Model	Accuracy (%)	Precision	Recall	F1-score	ROC-AUC score
LR	0.84	0.86	0.89	0.83	0.9
DT	0.93	0.94	0.94	0.92	0.93
RF	0.98	1	0.95	0.975	0.98
SVM	0.85	0.81	0.93	0.87	0.85
XGB	0.97	0.987	0.95	0.97	0.97
KNN	0.86	0.79	1	0.85	0.96

Table 4 presents the code for the tuned values with respect to the two tuning algorithms. Figure 7 presents the confusion matrices for the usual implementation of the models for the ML models. Figures 8 and 9 present the confusion matrices for the tuned models using *RandomisedSearchCV* and *GridSearchCV*.

Table 4 Hyper-parameter values for the two tuning algorithms

Model	RandomizedSearchCV	GridSearchCV
Logistic regression	{'class_weight': 'balanced', 'dual': False, 'max_iter': 250, 'penalty': 'l2'}	{'class_weight': 'None', 'dual': False, 'max_iter': 250, 'penalty': 'none'}
Decision tree classifier	{'criterion': 'entropy', 'max_depth': 560, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2}	{'criterion': 'entropy', 'max_depth': 560, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2}
Random forest classifier	{'criterion': 'gini', 'max_depth': 230, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 1400}	{'bootstrap': True, 'max_depth': None, 'max_features': 'auto', 'n_estimators': 11}
SVM	{'C': 1000, 'degree': 3}	{'C': 1000, 'degree': 5, 'kernel': 'poly'}
XGBOOST	{'colsample_bylevel': 0.7, 'colsample_bytree': 0.8, 'gamma': 0, 'learning_rate': 0.2, 'max_depth': 15, 'min_child_weight': 0.5, 'n_estimators': 100, 'reg_lambda': 1.0, 'silent': False, 'subsample': 0.5}	{'colsample_bytree': 0.5, 'gamma': 0, 'learning_rate': 0.1, 'max_depth': 7, 'reg_lambda': 10, 'scale_pos_weight': 3, 'subsample': 0.8}
ADABOOST	{'learning_rate': 1.0, 'n_estimators': 50}	{'learning_rate': 0.1, 'n_estimators': 500}
GRADIENTBOOST	{'learning_rate': 0.15, 'n_estimators': 1500}	{'learning_rate': 0.05, 'n_estimators': 250}

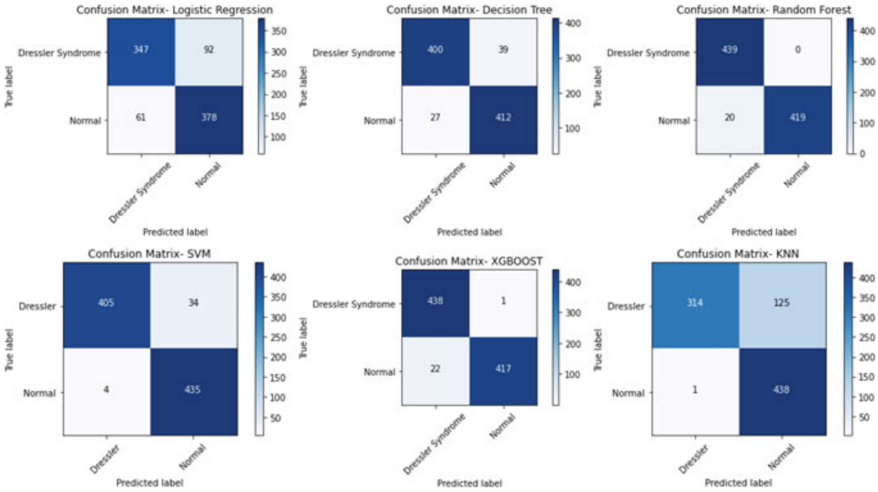

Fig. 7 Confusion matrices with respect to usual implementation

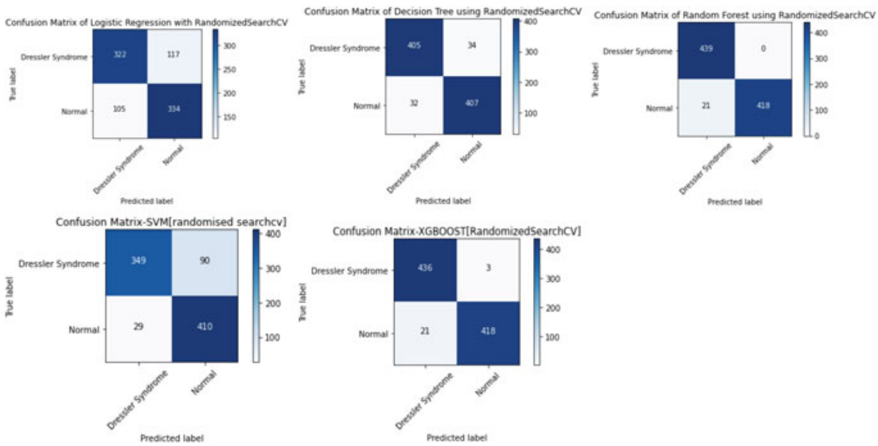

Fig. 8 Confusion matrices with respect to tuned implementation using *RandomisedSearchCV*

From the results analysis it is claimed that since Dressler syndrome is an outcome, hence it is deduced through this study that if symptoms for Dressler syndrome are observed, then it can be helpful in arresting myocardial infarction. The dataset used for this study comprised of 12 labeled values which can be correlated through 111 features.

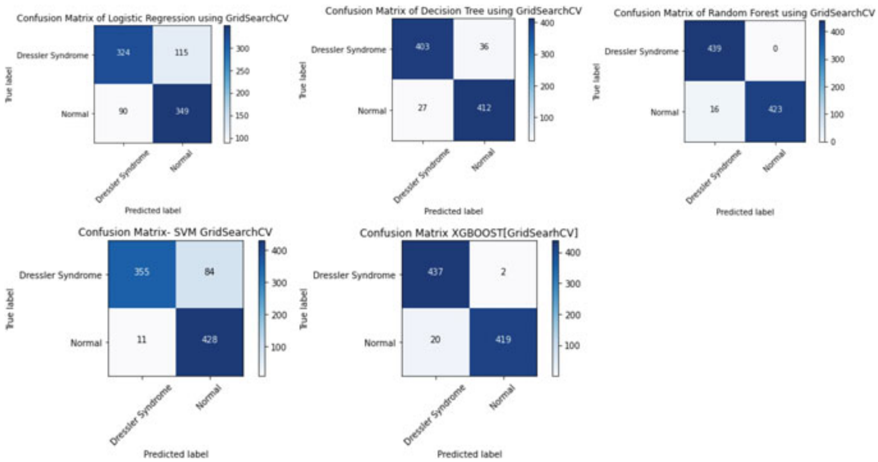

Fig. 9 Confusion matrices with respect to tuned implementation using *GridSearchCV*

5 Conclusion

This study presents the classification of Dressler syndrome using historical myocardial infarction data. In this study, we have used only one label (Dressler syndrome). The other 11 labels can be further classified in the future. Moreover, a uniform classification model can be generated which can classify all the 12 labels accurately using the myocardial infarction data. This study is the first of the twelve classifications based on 12 labels in the dataset.

References

1. Choudhary P, Dandapat S (2020) An evaluation of machine learning classifiers for detection of myocardial infarction using wavelet entropy and eigenspace features. In: 2020 IEEE applied signal processing conference (ASPCON), pp 222–226
2. Fatimah B, Singh P, Singhal A, Pramanick D (2021) Efficient detection of myocardial infarction from single lead ECG signal. Biomed Signal Process Control 68
3. Dressler's syndrome. Cleveland Clinic. https://my.clevelandclinic.org/health/diseases/17947-dresslers-syndrome. Accessed 2 May 2019
4. Mayo Clinic. https://www.mayoclinic.org/diseases-conditions/dresslers-syndrome/symptoms-causes/syc-20371811#:~:text=Dressler%20syndrome%20is%20a%20type,surrounding%20the%20heart%20(pericardium
5. Golovenkin, Shulman, Rossiev DA, Shesternya Myocardial infarction complications Data Set. https://archive.ics.uci.edu/ml/datasets/Myocardial+infarction+complications. Accessed 9 Dec 2020
6. Dohare A, Kumar V, Kumar R (2018) Detection of myocardial infarction in 12 lead ECG using support vector machine. Appl Soft Comput 64(1568–4946):138–147
7. Sun L, Lu Y, Yang K, Li S (2012) ECG analysis using multiple instance learning for myocardial infarction detection. IEEE Trans Biomed Eng 59:3348–3356

8. Ibrahim L, Mesinovic M, Yang K, Eid M (2020) Explainable prediction of acute myocardial infarction using machine learning and Shapley values. IEEE Access 8:210410–210417
9. Degerli A, Zabihi M, Kiranyaz S, Hamid T (2021) Early detection of myocardial infarction in low-quality echocardiography. IEEE Access 9:34442–34453
10. Sharma L, Sunkaria R (2018) Inferior myocardial infarction detection using stationary wavelet transform and machine learning approach
11. Hadanny A, Shouval R, Wu J, Shlomo N (2021) Predicting 30-day mortality after ST elevation myocardial infarction: machine learning-based random forest and its external validation using two independent nationwide datasets. J Cardiol 78(5):439–446
12. Tay D, Poh C, Reeth E, Kitney R (2015) The effect of sample age and prediction resolution on myocardial infarction risk prediction. IEEE J Biomed Health Inform 19(3):1178–1185
13. Kayyum S, Miah J, Shadaab A, lIslam M (2020) Data analysis on myocardial infarction with the help of machine learning algorithms considering distinctive or non-distinctive features. In: 2020 international conference on computer communication and informatics (ICCCI), pp 1–7
14. Omar N, Dey M, Ullah M (2020) Detection of myocardial infarction from ECG signal through combining CNN and Bi-LSTM. In: 2020 11th international conference on electrical and computer engineering (ICECE), pp 395–398
15. Martin H, Izquierdo W, Cabrerizo M, Cabrera A (2021) Near real-time single-beat myocardial infarction detection from single-lead electrocardiogram using long short-term memory neural network. Biomed Signal Process Control 68(1746–8094)
16. Bhaskar N (2015) Performance analysis of support vector machine and neural networks in detection of myocardial infarction. Procedia Comput Sci 46(1877–0509):20–30
17. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in Python 2825–2830