ILKOM Jurnal Ilmiah Vol. 15, No. 3, December 2023, pp. 413-426
Accredited 2nd by RISTEKBRIN No. 200/M/KPT/2020; E-ISSN 2548-7779 | P-ISSN 2087-1716

Research Article Open Access (CC–BY-SA)

Cloud-Based Realtime Decision System for Severity Classification of COVID-19 Self-Isolation Patients using Machine Learning Algorithms

Bhima Satria Rizki Sugiono[a,1]; Mokh. Sholihul Hadi[a,2,*]; Ilham Ari Elbaith Zaeni [a,3]; Sujito [a,4]; Mhd Irvan [b,5]
[a] *Department of Electrical Engineering and Informatics, Universitas Negeri Malang, Malang, 65145, Indonesia*
[b] *Graduate School of Information Science and Technology, The University of Tokyo, Japan*
[1] *Bhimasatria.1905366@students.um.ac.id;* [2]*mokh.sholihul.ft@um.ac.id;* [3]*ilham.ari.ft@um.ac.id;* [4]*sujito.ft@um.ac.id;*
[5] *irvan@yamagula.ic.i.u-tokyo.ac.jp*
* *Corresponding author*

Article history: Received October 23, 2023; Revised December 01, 2023; Accepted December 06, 2023; Available online December 20, 2023

Abstract
The global impact of the COVID-19 pandemic has been profound, affecting economies and societal structures worldwide. Indonesia, with a high caseload, has encountered significant challenges across various sectors. Virus transmission primarily occurs through physical contact, and the surge in active cases has strained hospital capacities, leading to the hospitalization of only severe cases. The remaining patients receive home telecare, but some experience sudden health deterioration with fatal consequences. To address this issue, this study proposes a remote outpatient care system utilizing Internet of Things (IoT) technology and medical electronics. This integrated system aims to provide an effective response to the COVID-19 pandemic. The research includes a comparative analysis of three machine-learning algorithms: decision tree, gradient tree boosting, and random forest for the classification of COVID-19 patients. The results reveal that the random forest algorithm outperforms the others with an accuracy rate of 70%, as compared to 67% for the decision tree and 62% for the gradient tree boosting algorithm. This integrated system not only addresses immediate healthcare delivery challenges but also offers data-driven insights for patient classification, thereby enhancing the effectiveness and reach of medical interventions.

Keywords: Decision Tree; Gradient Tree Boosting; Random Forest; Realtime Decision.

Introduction

The COVID-19 outbreak has caused an unprecedented level of difficulty for everyone on earth, putting an end to life as we know it and taking thousands of lives. The death toll from COVID-19 has reached 5, 212, 172, and 334,915 per May 22, 2020 because of the disease's spread [1], [2]. The lack of health facilities and medical personnel causes uneven monitoring activities for self-isolation patients. The need for a monitoring system that can monitor the condition of self-isolation patients with COVID-19. The system must be able to read the condition of patients such as body temperature, heart rate, and SpO2 levels in the blood, where these three parameters are several variables that indicate whether the patient is included in the light or severe disease category.

In addition to the system, it must be able to provide hospitalization recommendations for self-isolating patients based on all three parameters of their condition. Patients must be hospitalized and referred to hospitals. If patients in self-isolation experience a sudden decline in SpO2 levels, they should seek immediate medical attention. This incident can be avoided by implementing a monitoring system with Adaptive Cloud Learning and utilizing the Random Forest Classifier (RF), Decision Tree (DT), and Gradient Tree Boosting (GTB) for the Covid-19 Patient Inpatient Recommendation System, based on the preceding description.

The advantage of the designed system is that it employs machine learning algorithms with different approaches. The RF classifier method, a well-known ensemble method, is used to classify large amounts of data efficiently [3]. DT algorithm, on the other hand, utilizes a tree-like structure for decision making based on feature values. The GTB algorithm combines multiple decision trees sequentially to improve the overall predictive performance. The patient's self-isolation condition, as measured by body temperature, blood oxygen levels, and heart rate, can be used by all three algorithms to determine whether hospitalization is necessary. By leveraging the capabilities of RF, DT and GTB, it is expected that the system provides accurate and timely hospitalization recommendations. This, in turn, aims to decrease the mortality rate among self-isolated patients with deteriorating health conditions due to delayed treatment.

Several studies on telemedicine systems for covid, such as Gupta's Prediction of Confirmed Cases of COVID-19, Death, and Recovery in India, have been conducted modeling with RF. However, the study only predicted the death rate due to covid based on the characteristics of the date, time, location, recovered cases, death cases, and confirmed cases, so the patient's condition at the time was not directly considered [3]. Another study is the covid-19 diagnosis system accessible via the Heg.IA website, which is based on artificial intelligence [4]. Using a RF algorithm with 41 classes and 90 decision trees, the system aims to support decision-making regarding the diagnosis of Covid-19 and indications for hospitalization in regular, semi-ICU, and ICU wards. The system's weakness is that users must access the website and manually enter some data to be used as a decision-making aid, so that the detection of whether the user requires hospitalization is not in real time.

In Indonesia, extensive research has been conducted on the COVID-19 disease classification system, such as a study on the Coronavirus Disease 19 (COVID-19) Symptom Classification system using four machine learning algorithms: K-NN, Neural Network, RF, and Naive Bayes. The random forest algorithm achieved an accuracy of 68% in this study, allowing for the conclusion that random forests can be used in the classification process. The lack of a wearable device for real-time monitoring of the patient's condition is one of the deficiencies of the system designed in the study [5].

A system was designed to monitor the condition of COVID-19 self-isolation patients using a recommendation algorithm based on the RF, DT and GTB algorithms. Random forests are commonly used for classification tasks due to their ability to handle large datasets with reasonable accuracy [6]-[11]. Random forests are an ensemble of decision trees organized in a hierarchical structure. They are known for their ability to handle high-dimensional data and can effectively identify the most relevant attributes for classification [12], [13].

DTs are another popular algorithm for classification. They use a tree-like model of decisions and their possible consequences. However, DT can sometimes overfit the training data and may not generalize well to unseen data. Gradient tree boosting, on the other hand, is a boosting algorithm that combines multiple weak decision trees to create a strong predictive model. It iteratively builds trees to correct the errors made by previous trees. In the context of the system for monitoring COVID-19 self-isolation patients, the performance of the DT, GTB, and RF algorithms can be compared. RF is suitable for handling high-dimensional data, selecting relevant features, and providing accurate classification results. However, the DT and GTB algorithms can also be considered as alternative approaches, each with its own strengths and limitations, to determine the most suitable algorithm for the specific task at hand.

Method

This study involved the development of both hardware and software systems. The hardware related to telemedicine devices is used to detect the condition of COVID-19 self-isolation patients. The software, which in the form of machine learning installed in the cloud or on a server, is useful for making decisions on whether the patient needs hospitalization or not, focus on the research in this article that pertains to the algorithm's design. Among the methods used in this study are

A. Data Preparation

The dataset used in this research was obtained from a hospital in Malang City, where the data collection was conducted by one of the researcher's collaborating doctors through direct observation of patients infected with the COVID-19 virus. At this stage, the IoT system had not been implemented as the model was still in the development phase. The IoT system is intended for monitoring and classifying patient conditions once the algorithmic model is fully designed, hence no IoT system was used during the data collection phase. From the observations, 200 patient data points were collected and presented across four columns. This dataset undergoes a cleansing process, and classes that most significantly impact the condition of confirmed COVID-19 patients will be identified. Attributes such as respiratory issues, fever, and SpO2 levels are among those used. These three attributes critically affect the health of COVID-19 patients, and thus, employing these attributes, it is anticipated that the algorithm will be able to detect the condition of COVID-19 patients effectively. In the submitted manuscript, we delineate an algorithm specifically designed for classification purposes. The dataset employed in this study has been meticulously curated, drawing from authentic patient conditions, with data manually sourced from a hospital setting. Subsequent to the rigorous validation of the algorithm, it is intended to be integrated into a cloud-based infrastructure, facilitating real-time decision-making processes. For practical deployment, it is requisite for patients to utilize a wearable device, enabling continuous monitoring by healthcare professionals from remote locations through an Internet of Things (IoT) system. The detailed specifications and operational aspects of this wearable device will be comprehensively addressed in a forthcoming publication.

At the stage of data collection, the Knowledge Discovery in Database (KDD) method is employed, which is useful for extracting data from the database in order to identify patterns of patients infected with covid-19 by examining downloaded data patterns [14] – [16]. **Figure 1** depicts a KDD data retrieval process flow or stage.

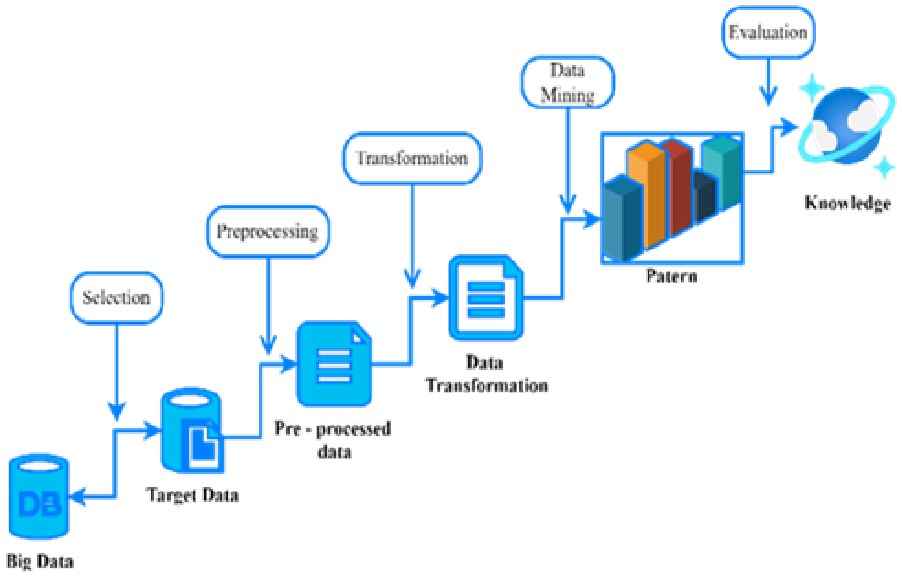

Figure 1. Deep stages in the KDD method

The stages of KDD method implementation are outlined as follows:

- Selection: The process of selecting relevant data in accordance with the research objectives.
- Pre-processing: Used for data cleansing (e.g., removing duplicate data or other issues) and to reproduce clean data prior to use.
- Transformation: The obtained data is then transformed into a feature of the selected data percentage in the form of code based on the information pattern.
- Data mining: pattern search by a particular method implies that this method is dependent on the tools and algorithms employed. Where in this study the random forest algorithm, decision tree, and gradient tree bosting was used.
- Pattern verification is the result of the testing procedure. It is performed to determine if the pattern matches expectations or if the results are contradictory.

In addition, the dataset used can be considered valid as it has been validated by the relevant doctors handling COVID-19 cases at the hospital. **Table 1** presents the data to be used in the study.

Table 1. Covid-19 Patient Data Attributes

Attribute	Data type	Condition
Heartbeat	Float	mild and severe
Fever	Float	
SpO2 Problems	Float	

Only three attributes from the dataset are selected for direct monitoring by the telemedicine device, out of the many available. The reason for selecting these three attributes is that they can be monitored in realtime, making them suitable parameters for real-time monitoring. These attributes provide valuable insights into the patient's condition and can be continuously monitored to assess any changes or deterioration in their health. By incorporating these realtime monitoring parameters into the algorithm, the system can provide immediate and accurate recommendations for the patient's condition and the need for hospitalization

B. Algorithm And System Design

In previous studies employing the RF algorithm, it was found that this algorithm could produce values with a high degree of accuracy [17]. This algorithm descends from ID3, which was created by J. Ross Quinlan [18]. Random Forest has the benefits of high performance, small error formation, and increased training [19]. In addition to the Random Forest classifier algorithm, this study also explores the DT and GTB algorithms for the classification of COVID-19 self-isolation patients. These algorithms offer their respective advantages in decision-making and predictive performance.

This research was conducted by developing a system of hospitalization recommendations for COVID-19 patients capable of classifying patients into two groups: those who require hospitalization and those who do not. **Figure 2** illustrates the implementation phase of the random forest classifier algorithm, as well as the decision tree and gradient tree boosting algorithms, on the given dataset.

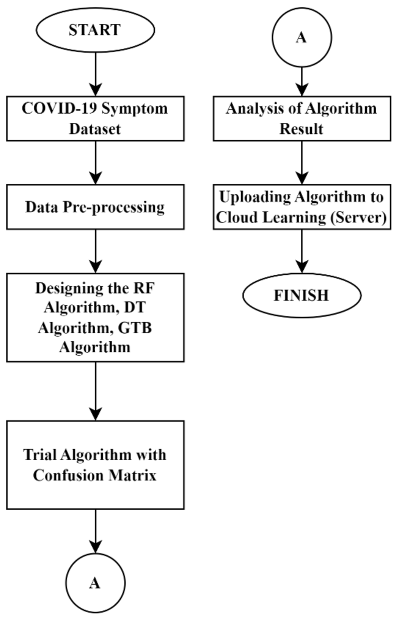
START
COVID-19 Symptom Dataset
Data Pre-processing
Designing the RF Algorithm, DT Algorithm, GTB Algorithm
Trial Algorithm with Confusion Matrix
A
A
Analysis of Algorithm Result
Uploading Algorithm to Cloud Learning (Server)
FINISH

Figure 2. Algorithm design flowchart

Creating an algorithm that can be used to classify the condition of COVID-19 patients requires multiple steps. The first step is data pre-processing, which involves the selection of independent and dependent variables in order to change the data type based on the requirements of the system, followed by the separation of the data into training data and test data. The ratio of test data to training data in the shared dataset is 30% for test data and 70% for training data [20]-[24]. The practice data is useful for obtaining values for the probability table, whereas the test data is used to validate the practice-generated probability tables.

After the data has processed, the implementation of three algorithms, namely RT, DT and GTB, is continued. These algorithms were chosen for their respective advantages. The RT algorithm, known for its ensemble nature, generates multiple decision trees based on a predetermined dataset and makes decisions based on these trees [25]-[28]. The DT algorithm utilizes a tree-like structure for decision making based on feature values, offering interpretability and simplicity in the decision-making process. The GTB algorithm combines multiple decision trees sequentially to improve overall predictive performance.

Following the successful development of the algorithm, the confusion matrix is used to evaluate the performance of each algorithm [29]-[33]. The final step is to analyze and draw conclusions regarding the performance of the designed algorithm after testing. If the algorithm, specifically the RT algorithm, demonstrates high precision, its design will be uploaded to the cloud for use in cloud-based learning for the classification of COVID-19 self-isolation patient conditions. By employing multiple algorithms, we aim to compare their performance and identify the most effective one for this classification task.

In the design of the DT algorithm, this study uses the entropy equation, making it appropriate to say that the decision tree employed in this research is based on the C4.5 model. The equation used to calculate the root node in the decision tree is as follows, as shown in Equation 1.

$$Entropy\ (S) = -\sum_{i=1}^{n} p_i log_2\, p_i \tag{1}$$

Where, p_i is the ratio between the number of samples in class I and the total number of samples in the dataset.

In addition, to evaluate each attribute in the C4.5 model, information gain is also used. Information gain is a measure to calculate how well an attribute separates a dataset based on its classes. It can be concluded that Information gain provides an assessment of the extent to which an attribute can provide useful information for classification. The general equation for Information Gain is as follows, as shown in Equation 2.

$$IG\ (S, A) = Entropy\ (S) - \sum \frac{|S_v|}{|S|} Entropy\ (S_v) \tag{2}$$

Where,

S	: Total number of data
A	: The attribute that will be evaluated
$\lvert S_v \rvert$	: Number of samples for v value
$\lvert S \rvert$	: The number of all data samples

In addition, the RT algorithm also employs the C4.5 decision tree model in its construction. Therefore, to evaluate each of its models, Equations 1 and 2 are also used. However, the steps in the RF algorithm differ, as can be seen in the pseudocode in the **Table 2**.

Table 2. Pesudocode for random forest algorithms

Algorithm: RandomForest_with_C4.5

```
Input: Dataset D, Number of trees N, Number of features to select m
Output: Random Forest Model

1. Initialize an empty list, Forest[]
2. For i = 1 to N do:
   a. Perform Bagging:
      1. Randomly sample with replacement 164 instances from D, let this new dataset be D_i.

   b. Select Features:
      1. Randomly select m features from the available features (e.g., body temperature, heart rate, SpO2 levels).

   c. Build Decision Tree with C4.5:
      1. Build a decision tree T_i using dataset D_i and selected features based on Information Gain, following the
         C4.5 algorithm.

   d. Append T_i to Forest[]

3. End For

4. For a new instance x, do:
   a. Initialize an empty list, Votes[]
   b. For each tree T_i in Forest do:
      1. Classify x using T_i
      2. Append the classification result to Votes[]
   c. The final prediction for x is the majority class in Votes[]

5. Return Forest as the final Random Forest Model
```

In addition to the RF model, this research also employs the GTB model. To construct the GTB algorithm model, Equation 3 is used, which represents the loss function.

$$L(y, F(x) \tag{3}$$

Where,

y	: The actual label
$F(x)$	: The model's prediction.

In the GTB model, the loss function is used as a metric to measure how well the model predicts the labels correctly. To construct a GTB model for classification, initially define an appropriate loss function, such as the logistic function, and initialize the initial model $F_0(x)$ based on the majority class of the training data. Subsequently, iterate through a series of DT. In each iteration, calculate the pseudo-residuals based on the gradient of the loss function, train a new decision tree $h_t(x)$ using these residuals, and update the model $F_t(x) = F_t - 1(x) + \nu h_t(x)$, where ν is the learning rate.

C. IoT System Architecture

The system is designed to classify the condition of COVID-19 patients using cloud computing. Cloud computing functions as Platform as a Service (PaaS) [34]-[36], which is responsible for analyzing all incoming data from mobile devices and offering services in the form of back end and front-end applications. **Figure 3** depicts the system's architecture from the wearable device to the cloud.

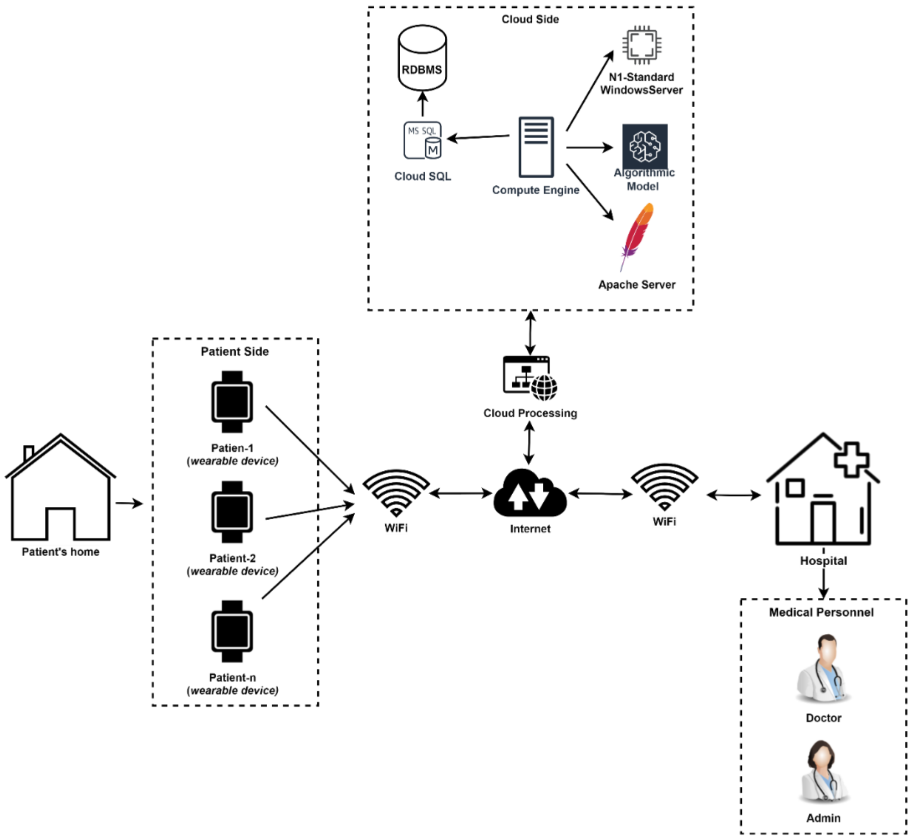

Figure 3. IoT System Architecture

In the design of IoT architecture, there are several user aspects, including the patient side, the medical personnel side, and the cloud processing side. On the patient side, there is a wearable device used by patients who are in their respective homes. The operating principle of the wearable device system itself is a device that automatically collects patient health data such as body temperature, heart rate, and blood oxygen levels. The results of the processed data are then sent to the cloud, where they will be analysed using pre-designed algorithms.

The wearable device will be used by patients in their homes, ensuring that they are fully monitored by medical personnel even though they are far from healthcare facilities. The wearable device itself is composed of several components; the microcontroller, serving as the core component of the system, is seamlessly integrated with three critical sensors in a bracelet-shaped wearable device: a heart rate sensor, a body temperature sensor, and an SpO2 sensor. Specifically, the research employs the MAX30120 sensor module, which is adept at monitoring both heart rate and SpO2 levels with high precision. This sensor configuration not only enhances the device's multifunctionality but also improves the reliability of the data. The sensor readings are wirelessly transmitted in real-time to a centralized server via an Internet-connected access point, ensuring timely data collection and analysis.

In the proposed cloud-based IoT system architecture, a Platform-as-a-Service (PaaS) environment is employed to offer an efficient and flexible cloud computing workspace. PaaS services enable users to configure their cloud computing environment according to specified requirements. Within this cloud system, a robust SQL-based database is utilized for the storage and management of extensive datasets. The stored data encompasses patient body temperature, heart rate, oxygen saturation, patient and medical personnel profiles, hospital addresses, and the classification outcomes from the designed machine learning algorithm models.

The machine learning algorithm side employs Python as the back-end language in the classification process. The constructed models utilize RT, DT and GTB algorithms for the meticulous analysis of sensor data, classifying the patient's condition into 'mild' or 'severe' symptom categories. This automated decision-making process is imperative for timely intervention and more effective healthcare management.

Additionally, the cloud processing side includes a monitoring website that can display patient and doctor data. The analysed patient data can be accessed by healthcare providers and related parties via a front-end application. This application, hosted on an Apache server and developed using HTML and PHP, offers a multiuser interface for patients at home as well as medical staff in hospitals [37], [38].

On the hospital side of this integrated IoT architecture, doctors and medical staff play a critical role in accessing and analysing monitoring data and the classification of patient conditions. They leverage the platform provided by the cloud system to access data that has been collected and analysed by the machine learning algorithms.

On the patient side, users, who may be patients themselves or family members, can actively monitor health activities read by the wearable device and view the current classification of the patient's condition. Within the patient interface, the website can display emergency contacts of the treating doctors, buttons to call an ambulance, and recommendations for referral hospitals based on the treating doctor's practice location.

The integration of the three components SQL database for data management, Apache for web services, and Python for machine learning algorithms creates a comprehensive cloud computing ecosystem, enabling real-time monitoring, analysis, and classification of health data.

The entire system is organized within a virtual machine named Google Compute Engine with an N1-Standard virtual machine architecture. The selection of the N1-Standard virtual machine type is predicated on the system requirements a cloud processing capability that can host a website and perform data analysis using Python while running an SQL database system.

This cloud based IoT system architecture plays a pivotal role in providing scalable, reliable, and accessible storage and computing resources. Moreover, the cloud computing architecture also enables smooth data flow between wearable devices and the web-based monitoring system.

Furthermore, the cloud framework allows for adaptive scaling of memory and computing power, ensuring efficient data processing. This cloud-based approach is integral to the system's capability to monitor health in real time and classify patient conditions, thereby optimizing the delivery of healthcare services and clinical decision-making.

However, the primary focus of this research is to construct machine learning algorithm models using RT, DT and GTB. These three algorithms will then be analysed, evaluated, and validated to determine which algorithm performs the best. Thus, the focus of this study is to compare the performance of these three algorithm models. Therefore, the performance of the IoT and cloud computing system will be discussed in subsequent research.

Results and Discussion

In this section, we will discuss the results of the conducted research and the Python-based model [39]-[41], that was used to analyze the data. There are 200 data points that have been generated from the pre-processed dataset, consisting of three attributes and two classes, namely 0 and 1. Class 0 corresponds to COVID-19 patients with mild symptoms, whereas class 1 corresponds to COVID-19 patients with severe symptoms. There are 108 data for class 0 and 92 data for class 1. The SMOTE technique is then used to rearrange data samples to improve the accuracy of the designed algorithm model [42]-[49].

A. Handling of Imbalanced Datasets (SMOTE)

This study's dataset included 200 COVID-19 patient records, which were separated into training data and test data by a ratio of 30% training data to 70% test data. Splitting the data is called the hold out method. One of the advantages of using this method is that hold out is a method that is simple and easy to understand. It also allows us to compare different models easily, because we only need to evaluate the model against the same test date. There is a disparity between data with mild and severe symptoms, as determined by medical personnel's collection of data on a variety of conditions. **Table 3** depicts the condition of the dataset prior to SMOTE-based oversampling. It is essential to reemphasize that the dataset utilized in this study was directly obtained from a hospital in Malang City. The data collection was conducted through direct observation by a collaborating doctor of the research team, spanning from July 2021 to August 2021. From this observation, a total of 200 datasets were acquired, which were subsequently divided into 30% (60 data) for testing and 70% (140 data) for training. The divided datasets were utilized to train and validate the model. The outcome of this training will be implemented in a real-time decision-making process within an IoT-based system for classifying the condition of COVID-19 patients. The algorithmic model will be embedded in the cloud, and body temperature sensors, SpO_2, and heart rate monitors will be employed to assess the patient's condition. The readings from these sensors will be transmitted to the cloud and analyzed by the developed algorithmic model, thereby yielding a classification of the patient's condition as either favorable or adverse.

Table 3. Number of Rehearsal Data Before Oversampling

Class	Sum
0	82
1	58

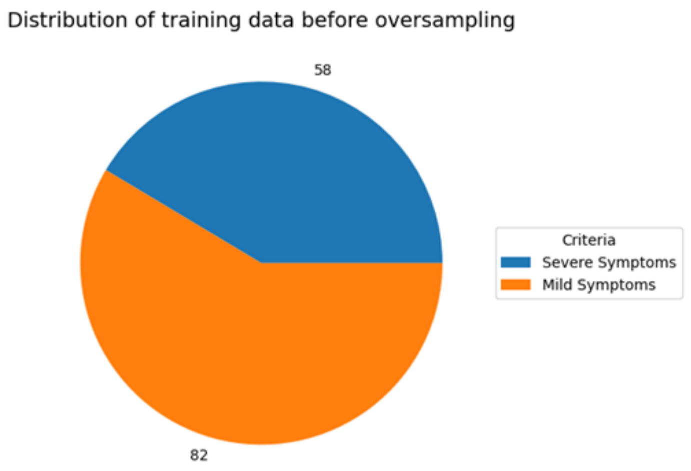

Figure 4. Distribution of training data before oversampling

Table 3 displays the datasets prior to SMOTE oversampling, and it shows an imbalance with 82 datasets for class 0 and 58 datasets for class 1. In this study, class 0 represents COVID-19 conditions with mild symptoms, while class 1 represents the criteria for conditions with severe symptoms. Since the imbalanced dataset can lead to an inaccurate random forest algorithm model, oversampling is necessary. To address this, the SMOTE is employed to add synthetic datasets to the minority class and achieve a more balanced dataset. By using SMOTE, the dataset is augmented with synthetic examples, which helps improve the performance of the classification algorithm and mitigate the bias towards the majority class.

After oversampling, the data above now has a balanced distribution, as can be seen in **Table 4**. The oversampling process has effectively increased the number of samples in the minority class, resulting in a more balanced dataset. This helps address the issue of class imbalance and ensures that the classification model can learn from enough instances from both classes, improving its ability to accurately classify COVID-19 conditions with varying symptom severity.

Table 4. Number of Rehearsal Data After Oversampling

Class	Sum
0	82
1	82

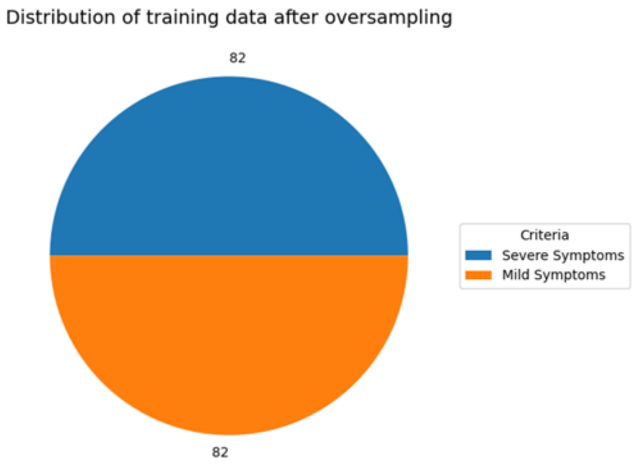

Figure 5. Distribution of training data before oversampling

After oversampling, the data now has a balanced distribution, as can be seen in Table 4. The number of samples for class 1 or severe symptoms has increased from 58 to 82 after oversampling. Originally, there were only 58 samples for this class. With the dataset now balanced, the next step is to train and test the algorithm model to evaluate its performance in accurately classifying COVID-19 conditions with varying symptom severity.

B. Testing of The Random Forest Model

The trial of this model was conducted under two conditions, the conditions before oversampling and the conditions after oversampling. Several analyses, including accuracy, precision, recall, and the f1 score, will be conducted to determine the performance of the developed model. Before oversampling, the data from model trials are displayed in Table 5.

Table 5. Model Trial Result Before Oversampling

Class	Precision	Recall	F1-score	Accuracy
0	0.62	0.81	0.70	70%
1	0.81	0.62	0.70	

The precision value for class 0 or mild symptoms, which is a prediction for what percentage of mild symptoms are correct mild symptoms from the overall data of patients predicted symptoms, yields a value of 62%, as displayed in Table 4. As for class 1 or severe symptoms, which is a prediction for what percentage of severe symptoms are accurate severe symptoms based on the overall data of predicted severe symptoms for patients, the value is 81%. It returns a value of 81% for the recall value in class 0 while it returns a value of 62% for class 1. Callous produces a value of 70% for the second F1-score result. As for the accuracy of the RF algorithm model without oversampling, it generates a value of 70%. The next test is a test run of the model after oversampling. Table 6 shows the model test result data.

Table 6. Model Trial Result After Oversampling

Class	Precision	Recall	F1-score	Accuracy
0	0.63	0.73	0.68	70%
1	0.77	0.68	0.72	

The designed model generates the following data after oversampling: It results in a trial value of 63% for a precision value in class 0. Regarding class 1, it returns the value 77%. Class 0 real values are returned with a value of 73%, while class 1 real values are returned with a value of 68%. Class 0 returns a value of 68% for Value F1, while class 1 returns a value of 72%. As a result of oversampling, the model's accuracy yields a value of 70%.

C. Testing of The Decision Tree Model

The second algorithm used is a DT, where two conditions will be tested: before and after oversampling. The analysis is conducted by examining the accuracy, precision, recall, and F1 score to assess the performance of the built model. **Table 7** demonstrates the performance of the decision tree algorithm before oversampling.

Table 7. Model Trial Result Before Oversampling

Class	Precision	Recall	F1-score	Accuracy
0	0.49	0.77	0.60	55%
1	0.68	0.38	0.49	

According to the results of the trial, the precision value of the DT method is 0.49, showing that 49% of the predictions classed as class 0 are correct, and 0.68, indicating that 68% of the predictions classified as class 1 are correct. Meanwhile, class 0 has a recall score of 0.77. This means that the model correctly identifies 77% of the samples that belong to class 0. As a result, the model correctly identifies 77% of the data that belong to class 0. The recall for class 1 is 0.38, meaning that the model can only reliably detect 38% of the samples that genuinely belong to class 1. In other words, the model can accurately categorize only 38% of all samples that genuinely belong to class 1.

Class 0's recall score, on the other hand, is 0.77. This shows that the model accurately identifies 77% of the samples that correspond to class 0. As a result, the model correctly classifies 77% of the samples that genuinely belong to class 0. Nevertheless, the recall for class 1 is 0.38, indicating that the model properly detects just 38% of all samples that genuinely belong to class 1. In other words, the model correctly classifies just 38% of the data that genuinely belongs to class 1. The accuracy of the decision tree model before oversampling is 55%, suggesting that the model accurately predicts only 55% of the test data. The decision tree is then tested with a dataset that has been oversampled. **Table 8** shows the decision tree model's test results after oversampling.

Table 8. Model Trial Result After Oversampling

Class	Precision	Recall	F1-score	Accuracy
0	0.59	0.77	0.67	67%
1	0.77	0.59	0.67	

After oversampling, the precision for class 0 is 0.59, meaning that 59% of the predictions classified as class 0 are correct. The precision for class 1 is 0.77, which means that 77% of the predictions classed as class 1 are correct. In terms of recall, the model achieves 0.77 for class 0, meaning that the model correctly identifies 77% of all samples that belong to class 0. In contrast, the recall for class 1 is 0.59, meaning that the model can detect 59% of all samples that belong to class 1. For both classes 0 and 1, the F1-score, which measures the balance of precision and recall, is 0.67. This shows that both courses have a reasonable mix of precision and recall. The model's overall accuracy is 0.67, meaning that it accurately predicts 67% of all samples in the testing dataset. The accuracy of the model's predictions represents their overall correctness.

D. Testing of The Gradient Tree Boosting

The third phase of testing entails assessing the performance of the GTB algorithm under two conditions, post-oversampling, and pre-oversampling datasets. Subsequently, after attaining the desired model outcomes, an analysis is conducted to examine the accuracy, precision, F1-score, and overall performance of the GTB algorithm. The performance evaluation of the algorithm prior to the oversampling procedure is presented in **Table 9**.

Table 9. Model Trial Result Before Oversampling

Class	Precision	Recall	F1-score	Accuracy
0	0.58	0.69	0.63	65%
1	0.72	0.62	0.67	

The precision scores obtained from the examination of the GTB algorithm show that it performs satisfactorily. The precision score for class 0 is 0.58, meaning that 58% of the predictions classed as class 0 are correct. Similarly, the precision score for class 1 is 0.72, indicating that the model accurately predicted 72% of the samples classified as class 1.

Meanwhile, the GTB approach provides a recall score of 0.69 for class 0, indicating that the model is capable of properly detecting 69% of all samples in class 0. Similarly, for class 1, it achieves a value of 0.62, suggesting that the model can detect 62% of all samples that genuinely belong to class 1.

In addition, we assess the F1-scores, which provide a balanced evaluation of the model's precision and recall. The F1-score for class 0 is 0.63, showing a harmonious balance between the model's accuracy in predicting and detecting class 0 data. The F1-score for class 1 is 0.67, indicating a similar balance in the model's performance for class 1 samples. In predicting and recognizing positive samples, a higher F1-score indicates a better balance of precision and recall. The Gradient Tree Boosting technique findings show that the model has an accuracy of 0.65, which means that it accurately predicts 65% of all samples in the testing dataset. Following that, we report the results of evaluating the GTB algorithm using the oversampling technique, which are shown in **Table 10**.

Table 10. Model Trial Result After Oversampling

Class	Precision	Recall	F1-score	Accuracy
0	0.55	0.62	0.58	62%
1	0.68	0.62	0.65	

Precision ratings for classes 0 and 1 were 0.55 and 0.68, respectively. This demonstrates that 68% of forecasts in the class 1 category are correct, compared to 55% of forecasts in the class 0 category. Recall ratings for classes 0 and 1 are 0.62 and 0.62, respectively. This means that the model can detect 62% of samples in class 0 and 62% of samples in class 1. F1 scores for classes 0 and 1 are 0.58 and 0.65, respectively. A higher F1-score indicates a better balance of memory and precision. In this case, the model performs reasonably well in both class 0 sample prediction and detection and class 1 sample prediction and detection. The model correctly predicted 62% of the samples in the dataset with an overall accuracy of 0.62.

Conclusion

Based on the data, it is possible to conclude that the RF model has an accuracy of 70% both before and after oversampling. Although there are differences in other measures such as precision, recall, and F1-score for various classes, these differences are minor and have little impact on the model's overall accuracy. Oversampling did not result in a substantial improvement in the performance of this model. While there was a minor increase in precision and recall for class 0 following oversampling, this was offset by a fall in recall for class 1. As a result, the accuracy of the Random Forest model did not change significantly after oversampling.

The decision tree model achieved an accuracy of 55% prior to oversampling, implying that it correctly predicted the class labels of 55% of the test samples. This suggests that the model's performance was moderate, as it struggled to predict accurately for a significant portion of the data. However, after implementing oversampling, the DT model's accuracy increased to 67%, indicating a significant improvement in its predictive capabilities. The model was able to correctly classify a higher percentage of the test samples because of this improvement, demonstrating its increased effectiveness in capturing the underlying patterns and relationships in the data. For the Gradient Tree Boosting algorithm, the model's accuracy before oversampling is 65%, but it decreases to 62% after oversampling. Before oversampling, the model accurately predicts 65% of the total samples in the test dataset. After oversampling, there is a decrease in the model's accuracy to 62%. However, this accuracy still demonstrates a reasonably good level of accuracy in predicting the classes in the test dataset.

Although there are changes in other metrics such as precision, recall, and F1-score, these changes are not significant and do not have a drastic impact on the overall accuracy of the model. Based on the information provided, the RF model appears to be the most accurate of the three algorithms discussed. The Random Forest model maintains an accuracy of 70% before and after oversampling, which is higher than the decision tree (67% after oversampling) and GTB (62% after oversampling) models. The RF model consistently demonstrates a relatively higher level of accuracy in correctly predicting the class labels of the test samples. While the decision tree model's accuracy improves after oversampling, it still falls short of the RF model. In contrast, the GTB algorithm exhibits a decrease in accuracy after oversampling, making it the least accurate of the three models.

As a result, based on the data provided, the RF algorithm is the most effective and accurate model for the given task. Its accuracy stability and relatively higher performance compared to the other algorithms make it a better choice. Random forest was chosen to build this system because the algorithm has been proven to have better performance in classifying COVID-19 patients compared to DT and GTB algorithms.

Acknowledgement

This research is sponsored by PTUPT Ristek Dikti research grant from the Ministry of Research and Technology of the Republic of Indonesia under 034/E5/PG.02.00.PT/2022 contract number.

References

[1] M. B. Jamshidi, A. Lalbakhsh, J. Talla, Z. Peroutka, F. Hadjilooei, P. Lalbakhsh, et al., "Artificial Intelligence and COVID-19: Deep Learning Approaches for Diagnosis and Treatment," IEEE Access, vol. 1, pp. 1-1, 2020.

[2] Q. Li, X. Guan, P. Wu, X. Wang, L. Zhou, Y. Tong, et al., "Early transmission dynamics in Wuhan, China, of novel coronavirus–infected pneumonia," New England journal of medicine, vol. 1, pp. 1-1, 2020.

[3] 3 V. K. Gupta, A. Gupta, D. Kumar, A. Sardana, "Prediction of COVID-19 confirmed, death, and cured cases in India using random forest model," Big Data Mining and Analytics, vol. 4, no. 2, pp. 116-23, Feb. 2021.

[4] 4 V. A. D. F. Barbosa, J. C. Gomes, M. A. de Santana, C. L. de Lima, R. B. Calado, C. R. Bertoldo Junior, et al., "Covid-19 rapid test by combining a random forest-based web system and blood tests," Journal of Biomolecular Structure and Dynamics, vol. 1, pp. 1-20, 2021.

[5] S. Anggraini, M. Akbar, A. Wijaya, H. Syaputra, M. Sobri, "Classification of Symptoms of Coronavirus Disease 19 (COVID-19) Using Machine Learning," Journal of Software Engineering Ampera, vol. 2, no. 1, pp. 57-68, 2021.

[6] B. A. Goldstein, E. C. Polley, F. B. Briggs, "Random forests for genetic association studies," Stat Appl Genet Mol Biol., vol. 10, no. 1, pp. 32, July 2011.

[7] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, B. P. Feuston, "Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling," Journal of Chemical Information and Computer Sciences, vol. 43, no. 6, pp. 1947–1958, 2003.

[8] L. Alhusain, A. M. Hafez, "Cluster ensemble based on Random Forests for genetic data," BioData Mining, vol. 10, no. 1, 2017.

[9] D. Yuan, J. Huang, X. Yang, J. Cui, "Improved random forest classification approach based on hybrid clustering selection," 2020 Chinese Automation Congress (CAC), 2020.

[10] C. Zhan, Y. Zheng, H. Zhang, Q. Wen, "Random-forest-Bagging broad learning system with applications for COVID-19 pandemic," IEEE Internet of Things Journal, vol. 8, no. 21, pp. 15906-15918, 2021.

[11] H. Lan and Y. Pan, "A Crowdsourcing Quality Prediction Model Based on Random Forests," in Proc. IEEE/ACIS 18th Int. Conf. on Computer and Information Science (ICIS), 2019, pp. 315-319.

[12] V. Jackins, S. Vimal, M. Kaliappan, and M.Y. Lee, "AI-based smart prediction of clinical disease using random forest classifier and Naive Bayes," The Journal of Supercomputing, vol. 77, no. 5, pp. 5198-5219, 2021.

[13] M. Hemalatha, "A hybrid random forest deep learning classifier empowered edge cloud architecture for COVID-19 and pneumonia detection," Expert Systems with Applications, vol. 210, p. 118227, 2022.

[14] A.D. Salman, H.A.D. AL-farttoosi, and A.J. Kadhim, "Study impact of the latitude on Covid-19 spread virus by data mining algorithm," in Journal of Physics: Conference Series, vol. 1664, no. 1, p. 012109, Nov. 2020.

[15] 15 S. Yamasaki, K. Yaji, and K. Fujita, "Knowledge discovery in databases for determining formulation in topology optimization," Structural and Multidisciplinary Optimization, vol. 59, no. 2, pp. 595-611, 2019.

[16] A. Linsel, K. Bär, J. Haas, J. Hornung, M.D. Greb, and M. Hinderer, "GeoReVi: A knowledge discovery and data management tool for subsurface characterization," SoftwareX, vol. 12, p. 100597, 2020.

[17] V.R. Sari, F. Firdausi, and Y. Azhar, "Comparison of Arabica Coffee Quality Predictions using SGD, Random Forest and Naive Bayes Algorithms," Edumatic: Journal of Informatics Education, vol. 4, no. 2, pp. 1-9, Dec. 2020.

[18] A. Assegaf, M.A. Mukid, and A. Hoyyi, "Bank Health Analysis Using Local Mean K-Nearest Neighbor and Multi Local Means K-Harmonic Nearest Neighbor," Gaussian Journal, vol. 8, no. 3, pp. 343-5, Aug. 30, 2019.

[19] F.S. Pamungkas, B.D. Prasetya, and I. Kharisudin, "Comparison of Supervised Learning Classification Methods on Data Bank Customers Using Python," in PRISMA, Proc. of the 2020 National Seminar on Mathematics, Mar. 4, vol. 3, pp. 692-697.

[20] H. Liu and M. Cocea, "Semi-random partitioning of data into training and test sets in granular computing context," Granular Computing, vol. 2, no. 4, pp. 357-386, 2017.

[21] J. Zhao, Y. Zhang, X. He, and P. Xie, "Covid-ct-dataset: a ct scan dataset about covid-19," arXiv preprint arXiv:2003.13865, 2020.

[22] G. Varma, A. Subramanian, A. Namboodiri, M. Chandraker, and C.V. Jawahar, "IDD: A dataset for exploring problems of autonomous navigation in unconstrained environments," in Proc. IEEE Winter Conf. on Applications of Computer Vision (WACV), 2019, pp. 1743-1751.

[23] M.M. Islam, F. Karray, R. Alhajj, and J. Zeng, "A review on deep learning techniques for the diagnosis of novel coronavirus (COVID-19)," IEEE Access, vol. 9, pp. 30551-30572, 2021.

[24] K.H. Abdulkareem, M.A. Mohammed, A. Salim, M. Arif, O. Geman, D. Gupta, and A. Khanna, "Realizing an effective COVID-19 diagnosis system based on machine learning and IOT in smart hospital environment," IEEE Internet of Things Journal, vol. 8, no. 21, pp. 15919-15928, 2021.

[25] E.P. Doheny, M. Flood, S. Ryan, C. McCarthy, O. O'Carroll, C. O'Seaghdha, P.W. Mallon, E.R. Feeney, V.M. Keatings, M. Wilson, and N. Kennedy, "Prediction of low pulse oxygen saturation in COVID-19 using remote monitoring post hospital discharge," International Journal of Medical Informatics, 2022, p. 104911.

[26] W. Hou, Z. Zhao, A. Chen, H. Li, and T.Q. Duong, "Machining learning predicts the need for escalated care and mortality in COVID-19 patients from clinical variables," International Journal of Medical Sciences, vol. 18, no. 8, p. 1739, 2021.

[27] D. Assaf, Y.A. Gutman, Y. Neuman, G. Segal, S. Amit, S. Gefen-Halevi, N. Shilo, A. Epstein, R. Mor-Cohen, A. Biber, and G. Rahav, "Utilization of machine-learning models to accurately predict the risk for critical COVID-19," Internal and Emergency Medicine, vol. 15, no. 8, pp. 1435-1443, 2020.

[28] F. Tezza, G. Lorenzoni, D. Azzolina, S. Barbar, L.A.C. Leone, and D. Gregori, "Predicting in-hospital mortality of patients with COVID-19 using machine learning techniques," Journal of Personalized Medicine, vol. 11, no. 5, p. 343, 2021.

[29] I. Markoulidakis, I. Rallis, I. Georgoulas, G. Kopsiaftis, A. Doulamis, and N. Doulamis, "Multiclass Confusion Matrix Reduction Method and Its Application on Net Promoter Score Classification Problem," Technologies, vol. 9, no. 4, p. 81, 2021.

[30] M. Pourhomayoun and M. Shakibi, "Predicting mortality risk in patients with COVID-19 using machine learning to help medical decision-making," Smart Health, vol. 20, p. 100178, 2021.

[31] K. El Asnaoui and Y. Chawki, "Using X-ray images and deep learning for automated detection of coronavirus disease," Journal of Biomolecular Structure and Dynamics, vol. 39, no. 10, pp. 3615-3626, 2021.

[32] G.Y. Kim, J.Y. Kim, C.H. Kim, and S.M. Kim, "Evaluation of deep learning for COVID-19 diagnosis: impact of image dataset organization," Journal of Applied Clinical Medical Physics, vol. 22, no. 7, pp. 297-305, 2021.

[33] A. Pranolo and Y. Mao, "Cae-covidx: Automatic covid-19 disease detection based on x-ray images using enhanced deep convolutional and autoencoder," International Journal of Advances in Intelligent Informatics, vol. 7, no. 1, pp. 49-62, 2021.

[34] M. Sevi and İ. Aydin, "COVID-19 detection using deep learning methods," in Proc. 2020 International conference on data analytics for business and industry: way towards a sustainable economy (ICDABI), 2020, pp. 1-6.

[35] C. Mouradian, F. Ebrahimnezhad, Y. Jebbar, J.K. Ahluwalia, S.N. Afrasiabi, R.H. Glitho, and A. Moghe, "An IoT Platform-as-a-Service for NFV Based-Hybrid Cloud/Fog Systems," IEEE Internet of Things Journal, 2020, doi:10.1109/jiot.2020.2968235.

[36] Z. Li, Y. Zhang, and Y. Liu, "Towards a full-stack devops environment (platform-as-a-service) for cloud-hosted applications," Tsinghua Science and Technology, vol. 22, no. 01, pp. 1-9, 2017.

[37] S.J. Taylor, A. Anagnostou, T. Kiss, G. Terstyanszky, P. Kacsuk, N. Fantini, D. Lakehal, and J. Costes, "Enabling cloud-based computational fluid dynamics with a platform-as-a-service solution," IEEE Transactions on Industrial Informatics, vol. 15, no. 1, pp. 85-94, 2018.

[38] P. Dauni, M.D. Firdaus, R. Asfariani, M.I.N. Saputra, A.A. Hidayat, and W.B. Zulfikar, "Implementation of Haversine formula for school location tracking," in Journal of Physics: Conference Series, vol. 1402, no. 7, p. 077028, IOP Publishing, 2019.

[39] M.M. Khan, M.R. Amin, A. Al Mamun, and A.A. Sajib, "Development of Web Based Online Medicine Delivery System for COVID-19 Pandemic," Journal of Software Engineering and Applications, vol. 14, no. 1, pp. 26-43, 2021.

[40] İ.B. Cicek, İ. Sel, F.H. YAĞIN, and C. Colak, "Development of a Python-Based Classification Web Interface for Independent Datasets," Balkan Journal of Electrical and Computer Engineering, vol. 10, no. 1, pp. 91-96, 2021.

[41] S.J. Sidiq and M. Zaman, "Using Unpruned Decision Tree and Random Forest For Learning From Multi-Class Imbalanced Data," Think India Journal, vol. 22, no. 30, pp. 758-763, 2019.

[42] A. Ishaq, S. Sadiq, M. Umer, S. Ullah, S. Mirjalili, V. Rupapara, and M. Nappi, "Improving the prediction of heart failure patients' survival using SMOTE and effective data mining techniques," IEEE Access, vol. 9, pp. 39707-39716, 2021.

[43] A. Fernández, S. Garcia, F. Herrera, and N.V. Chawla, "SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year anniversary," Journal of Artificial Intelligence Research, vol. 61, pp. 863-905, 2018.

[44] A. Ishaq, S. Sadiq, M. Umer, S. Ullah, S. Mirjalili, V. Rupapara, and M. Nappi, "Improving the prediction of heart failure patients' survival using SMOTE and effective data mining techniques," IEEE Access, vol. 9, pp. 39707-39716, 2021.

[45] M. Mukherjee and M. Khushi, "SMOTE-ENC: A novel SMOTE-based method to generate synthetic data for nominal and continuous features," Applied System Innovation, vol. 4, no. 1, p. 18, 2021.

[46] 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.

[47] X. Tan, S. Su, Z. Huang, X. Guo, Z. Zuo, X. Sun, and L. Li, "Wireless sensor networks intrusion detection based on SMOTE and the random forest algorithm," Sensors, vol. 19, no. 1, p. 203, 2019.

[48] S.F. Abdoh, M.A. Rizka, and F.A. Maghraby, "Cervical cancer diagnosis using random forest classifier with SMOTE and feature reduction techniques," IEEE Access, vol. 6, pp. 59475-59485, 2018.

[49] K. Shah, H. Patel, D. Sanghvi, and M. Shah, "A comparative analysis of logistic regression, random forest and KNN models for the text classification," Augmented Human Research, vol. 5, no. 1, pp. 1-16, 2020.