

## Article

# Optimizing HCV Disease Prediction in Egypt: The hyOPTGB Framework

Ahmed M. Elshewey <sup>1,\*</sup>, Mahmoud Y. Shams <sup>2</sup>, Sayed M. Tawfeek <sup>3</sup>, Amal H. Alharbi <sup>4</sup>,  
Abdelhameed Ibrahim <sup>5</sup>, Abdelaziz A. Abdelhamid <sup>6,7</sup>, Marwa M. Eid <sup>3,8,\*</sup>, Nima Khodadadi <sup>9</sup>,  
Laith Abualigah <sup>10,11,12,13,14,15,16</sup>, Doaa Sami Khafaga <sup>4</sup> and Zahraa Tarek <sup>17</sup>

- <sup>1</sup> Computer Science Department, Faculty of Computers and Information, Suez University, Suez 43533, Egypt
  - <sup>2</sup> Faculty of Artificial Intelligence, Kafrelsheikh University, Kafrelsheikh 33516, Egypt
  - <sup>3</sup> Department of Communications and Electronics, Delta Higher Institute of Engineering and Technology, Mansoura 35111, Egypt
  - <sup>4</sup> Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
  - <sup>5</sup> Computer Engineering and Control Systems Department, Faculty of Engineering, Mansoura University, Mansoura 35516, Egypt
  - <sup>6</sup> Department of Computer Science, Faculty of Computer and Information Sciences, Ain Shams University, Cairo 11566, Egypt
  - <sup>7</sup> Department of Computer Science, College of Computing and Information Technology, Shaqra University, Shaqra 11961, Saudi Arabia
  - <sup>8</sup> Faculty of Artificial Intelligence, Delta University for Science and Technology, Mansoura 35712, Egypt
  - <sup>9</sup> Department of Civil and Architectural Engineering, University of Miami, Coral Gables, FL 33146, USA; nima.khodadadi@miami.edu
  - <sup>10</sup> Computer Science Department, Prince Hussein Bin Abdullah Faculty for Information Technology, Al al-Bayt University, Mafrqa 25113, Jordan
  - <sup>11</sup> Department of Electrical and Computer Engineering, Lebanese American University, Byblos 13-5053, Lebanon
  - <sup>12</sup> Hourani Center for Applied Scientific Research, Al-Ahliyya Amman University, Amman 19328, Jordan
  - <sup>13</sup> MEU Research Unit, Middle East University, Amman 11831, Jordan
  - <sup>14</sup> Applied Science Research Center, Applied Science Private University, Amman 11931, Jordan
  - <sup>15</sup> School of Computer Sciences, Universiti Sains Malaysia, Gelugor 11800, Malaysia
  - <sup>16</sup> School of Engineering and Technology, Sunway University Malaysia, Petaling Jaya 27500, Malaysia
  - <sup>17</sup> Computer Science Department, Faculty of Computers and Information, Mansoura University, Mansoura 35561, Egypt
- \* Correspondence: ahmed.elshewey@fci.suezuni.edu.eg (A.M.E.); mmm@ieee.org (M.M.E.)



**Citation:** Elshewey, A.M.; Shams, M.Y.; Tawfeek, S.M.; Alharbi, A.H.; Ibrahim, A.; Abdelhamid, A.A.; Eid, M.M.; Khodadadi, N.; Abualigah, L.; Khafaga, D.S.; et al. Optimizing HCV Disease Prediction in Egypt: The hyOPTGB Framework. *Diagnostics* **2023**, *13*, 3439. <https://doi.org/10.3390/diagnostics13223439>

Academic Editor: Kathiravan Srinivasan

Received: 22 September 2023  
Revised: 4 November 2023  
Accepted: 8 November 2023  
Published: 13 November 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

**Abstract:** The paper focuses on the hepatitis C virus (HCV) infection in Egypt, which has one of the highest rates of HCV in the world. The high prevalence is linked to several factors, including the use of injection drugs, poor sterilization practices in medical facilities, and low public awareness. This paper introduces a hyOPTGB model, which employs an optimized gradient boosting (GB) classifier to predict HCV disease in Egypt. The model's accuracy is enhanced by optimizing hyperparameters with the OPTUNA framework. Min-Max normalization is used as a preprocessing step for scaling the dataset values and using the forward selection (FS) wrapped method to identify essential features. The dataset used in the study contains 1385 instances and 29 features and is available at the UCI machine learning repository. The authors compare the performance of five machine learning models, including decision tree (DT), support vector machine (SVM), dummy classifier (DC), ridge classifier (RC), and bagging classifier (BC), with the hyOPTGB model. The system's efficacy is assessed using various metrics, including accuracy, recall, precision, and F1-score. The hyOPTGB model outperformed the other machine learning models, achieving a 95.3% accuracy rate. The authors also compared the hyOPTGB model against other models proposed by authors who used the same dataset.

**Keywords:** hepatitis C virus (HCV); OPTUNA; hyperparameters; gradient boosting (GB); optimization

## 1. Introduction

Hepatitis C virus (HCV) is a blood-borne virus that primarily affects the liver, causing inflammation and potentially leading to cirrhosis, liver failure, and liver cancer [1,2]. HCV can be classified based on different factors, including the genotype, viral load, and disease stage. Genotype classification is based on the genetic makeup of the virus [3]. There are six major genotypes of HCV, each with several subtypes, and their distribution varies geographically. Genotype is essential in determining the course of treatment, as some genotypes are more responsive to certain medications than others. Viral load classification refers to the amount of virus in the blood [4]. High viral loads are associated with a greater risk of liver damage and may indicate a more aggressive disease course. Disease stage classification is based on the extent of liver damage caused by HCV. The disease can progress from a mild form of liver inflammation (chronic hepatitis) to more severe conditions of liver disease, such as cirrhosis and liver cancer. The stage of the disease is determined by evaluating liver function tests, imaging studies, and liver biopsy [5]. Overall, HCV disease classification plays a vital role in determining the appropriate course of treatment and monitoring disease progression. Early detection and treatment can help prevent or delay the development of severe liver complications associated with HCV infection [6]. In contrast to developed countries in Europe and North America, there is a higher incidence of hepatitis C virus (HCV) infection in the impoverished developing nations of Asia and Africa.

Moreover, countries like Pakistan, China, and Egypt have a larger population of individuals who suffer from chronic HCV infections [7,8]. The classification of HCV in Egypt is based on different factors, including genotype, viral load, and disease stage [7]. Genotype classification is based on the genetic makeup of the virus, and in Egypt, the most prevalent genotype is genotype 4. Viral load classification is based on the amount of virus present in the blood, and a high viral load is associated with a greater risk of liver damage. Disease stage classification is based on the extent of liver damage caused by HCV and can range from mild inflammation to severe liver diseases such as cirrhosis and liver cancer [9]. Healthcare workers in Egypt, who frequently interact with patients, are at an increased risk of HCV infection and other blood-borne diseases. Thus, it is crucial to understand the classification of HCV in Egypt to prevent and treat the disease effectively. Early detection and treatment of HCV can help prevent the development of severe liver complications associated with the disease [9].

There is a pressing need for accurate and dependable, non-invasive technology for the detection of HCV. Machine learning (ML) algorithms have effectively analyzed clinical data and identified intricate and nonlinear relationships in medical conditions. Classification techniques using ML algorithms can be employed to create models for diagnosing HCV by detecting infected individuals. However, unsuitable attributes in the attribute set can compromise the classifier's accuracy [10,11]. Hyperparameter tuning is a crucial step in building a machine learning model. It involves selecting the optimal values for the hyperparameters that determine the algorithm's behavior during training. Hyperparameters control various aspects of the model, such as the learning rate, regularization, number of layers, and number of nodes in each layer [12]. Choosing the right hyperparameters can significantly impact the model's performance, accuracy, generalization, and convergence speed. An appropriate selection of hyperparameters can improve the model's ability to generalize and make accurate predictions on unseen data. Conversely, inappropriate hyperparameters can lead to poor performance, overfitting, or underfitting. By optimizing the hyperparameters, we can improve the performance of the machine learning model, making it more accurate, robust, and efficient [13].

The term "optimization" refers to a vital instrument that is utilized in a variety of fields, one of which is the field of medicine, where it plays an important part. The goal of optimization is to arrive at the most advantageous results or choices attainable given a particular set of circumstances by making use of a variety of factors and predetermined standards. In the field of medicine, optimization is put to use in a wide variety of contexts,

including the following: the prediction and classification of monkeypox disease [14–16], the feature selection and classification in diagnosed breast cancer [17], the classification of diabetes [18], and the classification of COVID-19 in chest X-ray pictures [19]. The application of optimization in the field of medicine helps to improve the overall outcomes for patients and makes the most efficient use of the available resources.

The dataset used in this study is available at the UCI machine learning repository [20]. The dataset includes 1385 instances and 29 features, where 28 are predictors, and one is the target feature. The multi-class target feature, representing the baseline histological staging, contains instances with different values. Specifically, the values are portal fibrosis without septa, few septa, many septa without cirrhosis, and cirrhosis—the number of instances associated with values of 336, 332, 355, and 362, respectively.

Feature selection is a critical step in building a machine learning model. It involves selecting the most relevant and informative features from the available dataset to train the model. The goal of feature selection is to reduce the dimensionality of the dataset, simplify the model, and improve its accuracy and generalization ability [21]. Feature selection is essential for several reasons. First, it can improve the accuracy and efficiency of the model. By removing irrelevant or redundant features, we can reduce the noise in the dataset, improve the model's signal-to-noise ratio, and reduce overfitting. This, in turn, can improve the model's performance on new, unseen data. Second, feature selection can simplify the model and make it more interpretable. By removing irrelevant or redundant features, we can reduce the complexity of the model and make it easier to understand and explain. This is particularly important in applications where model interpretability is critical, such as healthcare, finance, and legal domains. Third, feature selection can reduce the cost and time required to train the model. Reducing the number of features can decrease the computational resources needed for training, testing, and deploying the model [22,23]. Numerous studies have proposed various methods for predicting HCV disease, and we have developed our system, hyOPTGB, which utilizes the gradient boosting (GB) model and OPTUNA hyperparameter tuning to make these predictions [24,25]. The GB model has become increasingly popular for classification problems due to its strong performance [26,27].

Efficient hyperparameter tuning is essential to improve the performance of any system. A hyperparameter is a configuration setting that is not learned from the data but is set prior to the training of a machine learning model. These settings are essential because they control various aspects of the model's training process and architecture, influencing its ability to learn and make predictions. Hyperparameters are distinct from model parameters, which are learned from the training data. Model parameters are the internal values that the machine learning algorithm adjusts during training to optimize the model's performance for a specific task. A good set of hyperparameters can significantly impact the prediction system's accuracy [28]. To achieve this, we chose to use OPTUNA for hyperparameter optimization, a widely used method that provides better optimization results. Each model has a unique set of hyperparameters and the goal is to find the best combination among them, a task known as hyperparameter optimization [29]. There are several ways to tackle this problem, including manual search, random search, grid search, and OPTUNA, which are considered the most effective. Manual tuning for a significant hyperparameter value is not recommended as it could be more efficient and effective [30]. Here are the key contributions of this paper:

1. We presented a robust hyOPTGB system, which utilizes optimized GB and OPTUNA to predict HCV disease.
2. The system's performance was enhanced through data preprocessing techniques, feature selection, and hyperparameter tuning.
3. GB is a distributed machine learning model based on trees renowned for its fast and efficient performance compared to other classification algorithms.
4. OPTUNA is an automated technique used for hyperparameter tuning, which helped optimize our system's performance.

The paper's structure is organized as follows: Section 2 will provide an overview of the related work carried out by other researchers. Section 3 will elaborate on the materials and methods employed in this study. Section 4 will analyze the results obtained. Lastly, Sections 5 and 6 will discuss and conclude the paper by summarizing the findings and proposing future research directions.

## 2. Related Work

Accurately determining the stage of liver fibrosis in patients with chronic hepatitis C (CHC) is critical for monitoring the disease, predicting treatment response, determining prognosis, and identifying the optimal treatment timing. However, a liver biopsy could be better due to its invasiveness, sampling errors, and high costs. To overcome these limitations, clinical information such as age, gender, body mass index (BMI), and non-invasive blood serum markers like alanine transaminase (ALT), aspartate transaminase (AST), glucose, hemoglobin (HGB), white blood cell (WBC) count, and red blood cell (RBC) count can be used to predict fibrosis stage. Various machine learning classification models have been applied in previous studies for HCV prediction, and this section discusses their results.

In a study by Tsvetkov et al. [24], a machine learning model was proposed to diagnose the stage of liver fibrosis in patients. The authors analyzed 1240 patient records with chronic viral hepatitis C and developed machine learning models using data from 689 patients classified by the stage of liver fibrosis. They identified critical predictors from nine widely used prognostic factors and obtained an accuracy of 80.56%. Akella et al. [25] conducted a study to create clinical risk models using machine learning algorithms to predict the extent of fibrosis in patients with chronic hepatitis C. They built nine ML algorithms using patient demographic information and standard serum laboratory values on an Egyptian cohort dataset. The extreme gradient boosting model achieved 81% precision in estimating fibrosis. The authors also found that most of their models performed better than existing diagnostic options in this patient group for assessing fibrosis.

The study conducted by Nandipati et al. [31] aimed to compare the performance of multiclass and binary class labels in predicting the hepatitis C virus using a dataset of Egyptian patients. They focused on identifying the essential features crucial in predicting the disease using different analytical tools. The results showed that a random forest using Python, and KNN using R had the highest precision rates of 54.56% and 51.06%, respectively, for both binary class and multiclass labels. Abd El-Salam et al. [32] used machine learning techniques to analyze a group of 4962 HCV patients in Egypt from 2006 to 2017. The study aimed to identify the presence or absence of esophageal varices in 2218 patients using 24 clinical laboratory variables. The researchers employed six popular classifiers: SVM, RF, C4.5, MLP, NB, and BN. The data were obtained from the Egyptian National Committee to Combat Viral Hepatitis, which manages the national treatment program for viral hepatitis patients in Egypt, overseen by the Ministry of Health. The study achieved accuracies ranging from 65.6% to 68.9% for the six classifiers.

Hashem et al. [33] utilized machine learning techniques to predict the development of hepatocellular carcinoma in patients with chronic liver disease due to HCV. They initially identified a group of input variables and then applied LR, DT, and CART algorithms to determine the optimal subset of variables. The findings revealed that the LR, DT, and CART algorithms achieved 96%, 99%, and 95.5% accuracy rates, respectively. Sartakhti et al. [34] proposed a new machine learning method that combines support vector machine (SVM) with simulated annealing (SA). They used a dataset from the UCI machine learning database and evaluated the classification accuracy using 10-fold cross-validation. The study found that the proposed approach achieved a high classification accuracy of 96.25%, surpassing other classification methods used for the same problem.

Vikas et al. [35] described a diagnostic system for identifying the presence of the hepatitis C virus (HCV). This system uses case-based reasoning (CBR) and correlation lift metrics to create a predictive model for detecting HCV. The proposed method can

accurately predict whether a patient's record shows HCV indications or not and can help reduce the risk factors associated with HCV in humans. Additionally, the system can differentiate between patient records of living and deceased individuals. Overall, this approach is highly effective in detecting HCV and can potentially decrease the virus's prevalence in the population. Zaki et al. [36] discussed how a rough set of data were examined to identify attribute dependency and to create a reduced set of attributes. The primary objective of the dataset was to predict whether HCV was present or absent. The analysis revealed that the proposed methodology was exact and achieved high accuracy in predicting HCV's presence or absence.

KayvanJoo et al. [37] conducted a study to predict the presence of HCV using machine learning techniques that analyze viral nucleotides. The research involved four approaches, DT, SVM, NB, and NN, to predict the response to interferon-alpha (IFN-alpha) and ribavirin (RBV) therapy based on processed features. The authors selected ten attribute weighting models from an initial dataset of 76 attributes, such as chi-square, Gini index, deviation, info-gain, info-gain ratio, SVM, PCA, uncertainty, relief, and rule. Finally, SVM, NB, NN, and DT were utilized for the classification process, with an average accuracy rate of 85%.

Until now, several methods have been employed to predict HCV disease prediction such as machine learning algorithms, feature selection methods, and deep learning methods. However, these methods have yet to be very effective in producing accurate predictions. To address this issue, this paper introduced a new model, hyOPTGB, which utilizes the GB model in combination with the trending hyperparameter optimization technique, OPTUNA. OPTUNA was chosen for its speed, efficiency, and proven success as one of the best available hyperparameter optimization techniques. Lai et al. [38] employed Optuna algorithm to optimize hyperparameters for forecasting models in this study. The findings of this investigation demonstrated that the utilization of Optuna in conjunction with five different tree-based machine learning models yielded highly satisfactory forecasting accuracy. Tonmoy et al. [39] introduced Optuna, an automated hyperparameter tuning algorithm, utilized to identify the best configurations for the dataset under investigation. The research introduces the architecture of the Optuna-Optimized GAN (OOG) method and showcases remarkable results, achieving accuracy, precision, recall, and F1 score.

Previous research studies failed to obtain good results due to their lack of hyperparameter optimization techniques. The hyOPTGB model aims to provide improved predictions for HCV disease prediction.

### 3. Materials and Methods

In this paper, we aim to improve the accuracy of predicting HCV disease among healthcare patients in Egypt by developing the hyOPTGB model. The system is designed to distinguish between HCV-infected patients and non-infected patients. Min-Max normalization is used to normalize the data; also, the forward selection (FS) wrapped method is used to select the most important features to increase the accuracy and efficiency of the classification process. To optimize the performance of the gradient boosting model, we fine-tuned its hyperparameters and then trained it using the resulting optimal parameters. We relied on the OPTUNA framework to carry out the hyperparameter tuning process. To evaluate the performance of our approach, we used a k-fold cross-validation ( $k = 15$ ) method on the training data. Our optimization goal is to improve the model's accuracy with each iteration. We have provided a visual representation of our proposed hyOPTGB model in Figure 1.

#### 3.1. Dataset

The dataset used in this study is available at the UCI machine learning repository [20]. The dataset includes 1385 instances and 29 features, where 28 are predictors, and one is the target feature. The multi-class target feature, representing the baseline histological staging, contains instances with different values. Specifically, the values are portal fibrosis without septa, few septa, many septa without cirrhosis, and cirrhosis—the number of instances



associated with values of 336, 332, 355, and 362, respectively. The description of the features is demonstrated in Table 1.

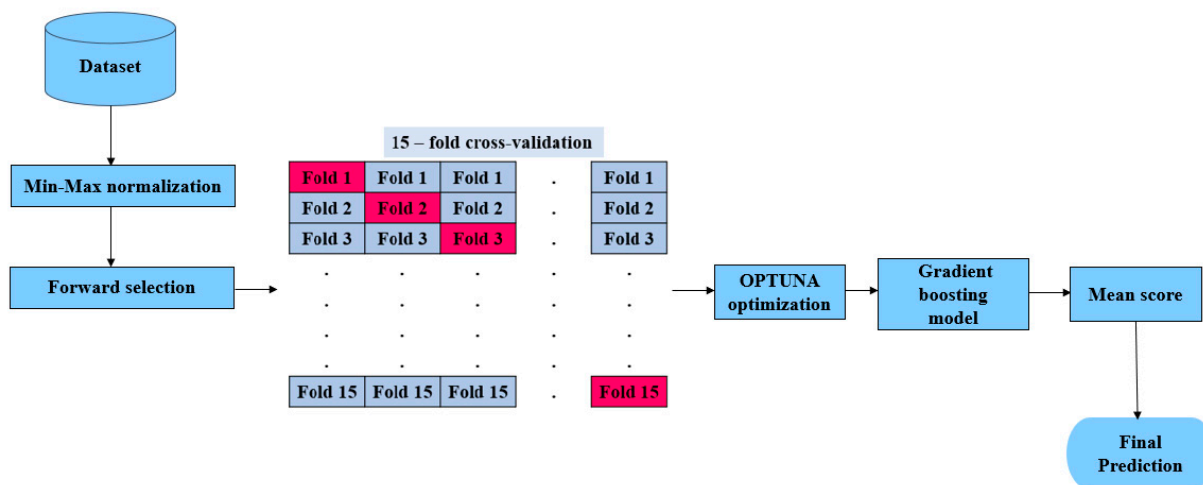


Figure 1. Representation of hyOPTGB model.

Table 1. Features description.

No.	Description	No.	Description
1	Age	16	ALT 1 (Alanine transaminase in 1 week)
2	Gender	17	ALT 4 (Alanine transaminase in 4 weeks)
3	BMI (Body mass index)	18	ALT 12 (Alanine transaminase in 12 weeks)
4	Fever	19	ALT 24 (Alanine transaminase in 24 weeks)
5	Nausea/Vomiting	20	ALT 36 (Alanine transaminase in 36 weeks)
6	Headache	21	ALT 48 (Alanine transaminase in 48 weeks)
7	Diarrhea	22	ALT after 24 w (Alanine transaminase after 24 weeks)
8	Fatigue & generalized bone ache	23	Base RNA (Ribonucleic acid)
9	Jaundice	24	RNA 4 (Ribonucleic acid in 4 weeks)
10	Epigastric pain	25	RNA 12 (Ribonucleic acid in 12 weeks)
11	WBC (White blood cell)	26	RNA EOT (Ribonucleic acid end of treatment)
12	RBC (Red blood cell)	27	RNA EF (Ribonucleic acid elongation factor)
13	HGB (Hemoglobin)	28	Baseline histological grading
14	Plat (Platelets)	29	Baseline histological staging (target feature)
15	AST 1 (Aspartate transaminase)		

The heatmap analysis for the dataset features is given in Figure 2. The heatmap analysis is a valuable tool for visualizing the correlation matrix of a dataset, where each element in the matrix represents the correlation coefficient between two variables. In machine learning, a heatmap can be used to identify which features in a dataset are strongly correlated with the target variable and each other. Figure 3 displays the box plot for the dataset features. A box plot, also known as a box-and-whisker plot, is a graphical representation of the distribution of a dataset. It displays the median, quartiles, and outliers of the data. In machine learning, a box plot can be used to identify outliers, visualize the spread of the data, and compare the distributions of different features.

### 3.2. Min-Max Normalization

Min-Max normalization is a widely used data preprocessing method in machine learning that normalizes the values of a feature to a specific range (typically between 0 and 1) by subtracting the minimum value and dividing it by the range of the data [40]. This technique aims to ensure that all features are uniformly scaled and enhance the specific algorithm’s accuracy. The steps for Min-Max normalization are as follows:

For a given feature of x:

1. Calculate the minimum value (min (x)) and maximum value (max (x)) of x across the dataset;
2. Subtract the minimum value from each value of x ( $x - \min(x)$ );
3. Divide the result by the range of the data ( $\max(x) - \min(x)$ ).

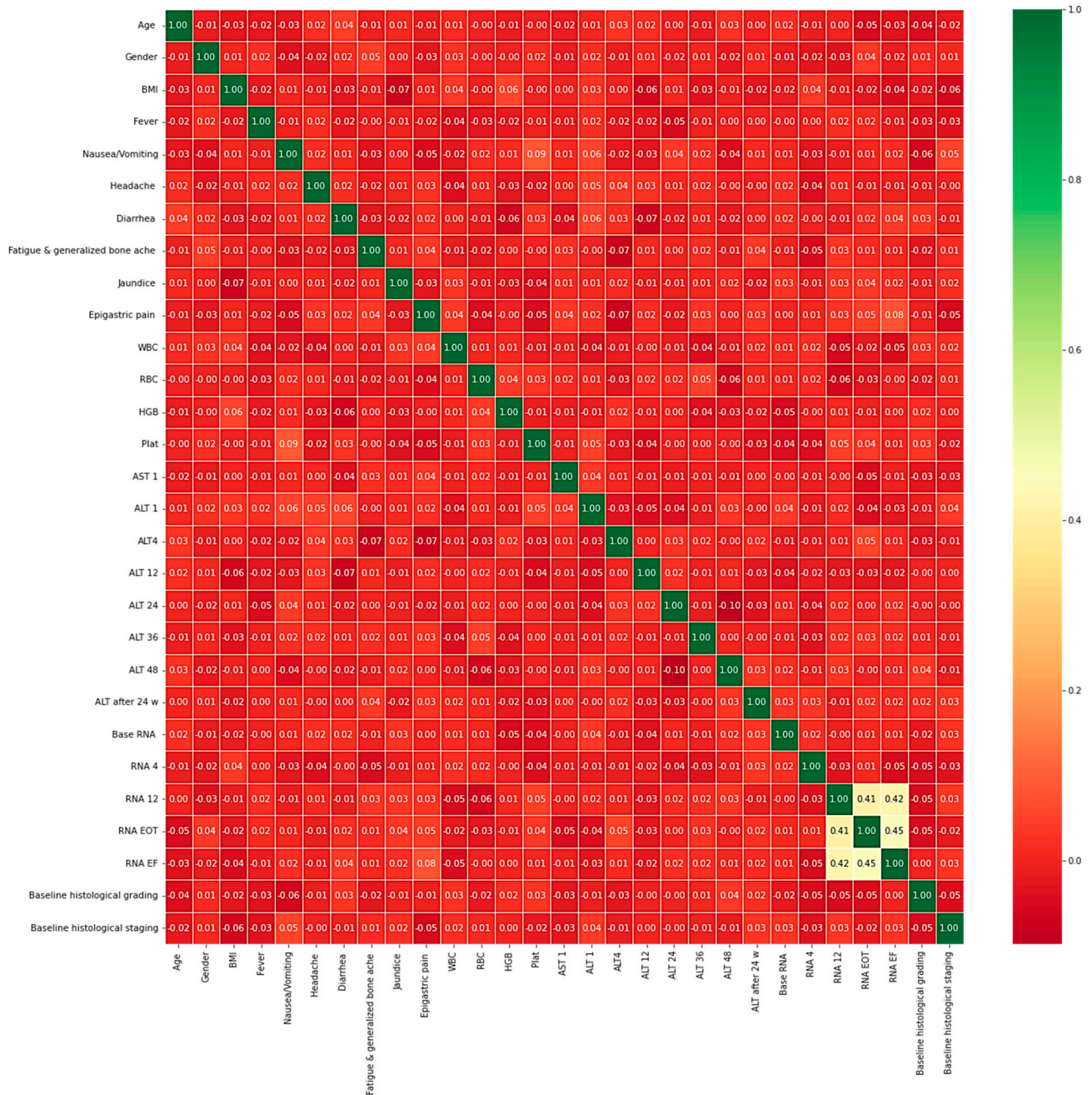


Figure 2. Heatmap analysis for the dataset features.

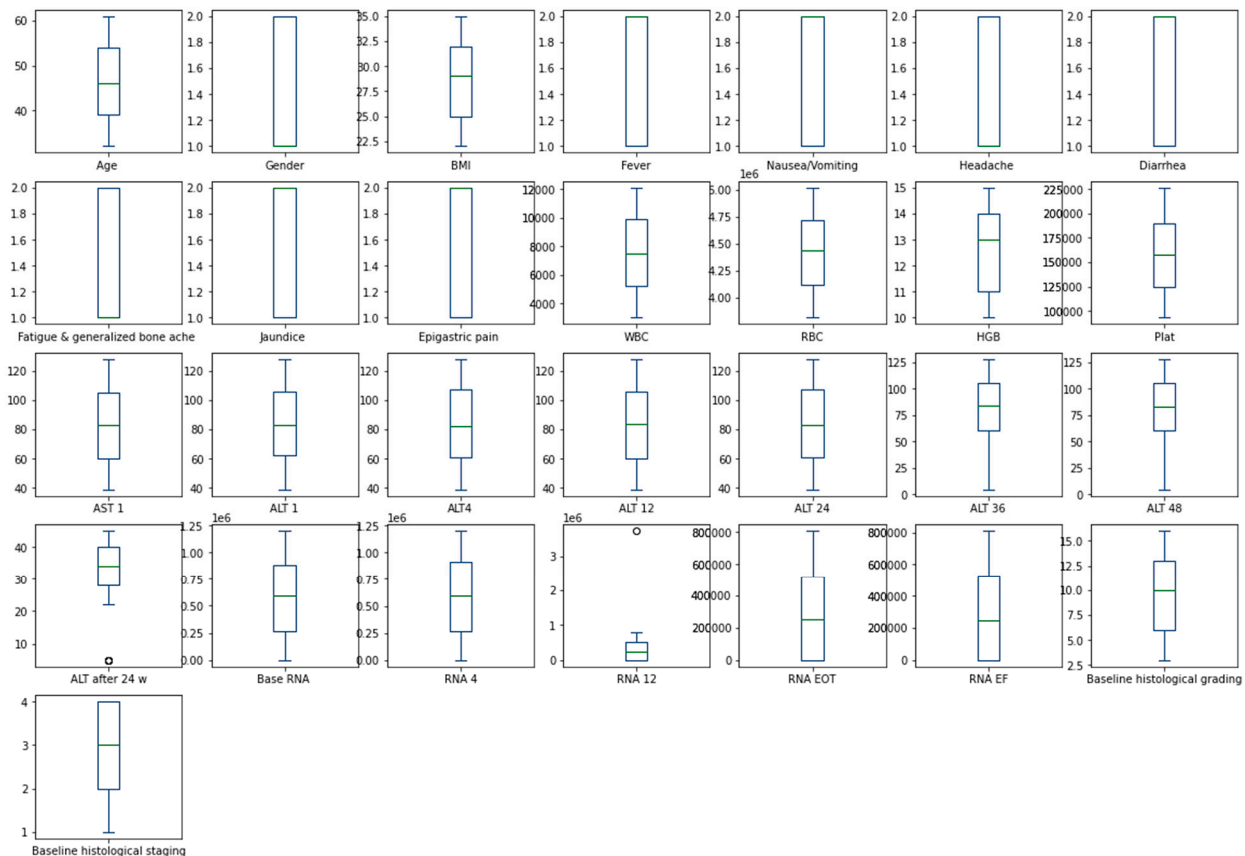
### 3.3. Forward Selection

Forward selection is a feature selection technique in machine learning where features are added to the model iteratively based on their performance until a stopping criterion is met [41,42]. The basic algorithm for forward selection can be expressed in the following steps:

1. Initialize the set of selected features S to be empty;
2. For each feature X not in S, train a model with the features in S union X and compute the model's performance;

3. Select the feature  $X$  that yields the best performance on the model trained with  $S$  union  $X$ , and add it to the set  $S$ ;
4. Repeat steps 2 and 3 until a stopping criterion is met (e.g., a predetermined number of features have been selected, the performance improvement falls below a certain threshold, etc.).

Upon conducting forward selection on the dataset utilized in this study, a set of 13 features were chosen. These features include BMI, fever, diarrhea, epigastric pain, WBC, HGB, plat, ALT 1, ALT 4, ALT 24, base RNA, RNA 4, and RNA EF.



**Figure 3.** Box plot for dataset features.

### 3.4. K-Fold Cross-Validation

K-fold cross-validation is a machine learning method used to assess a model's performance with a small amount of data [43]. It works by splitting the data into  $k$  subsets of equal size or "folds", using  $k - 1$  of them to train the model and holding out the remaining fold for validation. This process is repeated  $k$  times, with each fold used once for validation and the remaining folds used for training the model [44].

The steps involved in  $k$ -fold cross-validation are as follows:

1. Shuffle the dataset randomly;
2. Split the dataset into  $k$  groups of equal size;
3. For each group, select it as the validation set and use the remaining groups as the training set;
4. Train the model using the training set and evaluate it using the validation set;
5. Calculate the evaluation metric (e.g., accuracy, precision, recall, and F1-score.) for the model;
6. Repeat steps 3–5  $k$  times, using a different group as the validation set each time;
7. Calculate the average evaluation metric over the  $k$  repetitions.



K-fold cross-validation helps to reduce the risk of overfitting the model to the training data by allowing the model to be tested on different subsets of the data. It also helps to reduce the variance of the evaluation metric since it uses multiple validation sets rather than a single one.

### 3.5. OPTUNA Optimization

To improve the performance of machine learning models, proper hyperparameter tuning is necessary. This step is crucial in creating effective models as it significantly impacts the model's output. However, many individuals rely on trial and error, developing hyperparameters and testing them repeatedly over several hours or days. This approach could be more efficient and can be time-consuming. OPTUNA is a tool that automates the hyperparameter optimization process, eliminating the need for repetitive manual testing and saving valuable time and effort. OPTUNA is a tool that seeks to unify optimization paradigms by following a philosophy built upon four fundamental pillars.

#### 3.5.1. Design-by-Run API

The design-by-run API feature of OPTUNA enables the optimization of hyperparameters during the training process, eliminating the need for separate optimization runs and making the process more efficient. Design-by-run is a concept commonly used in deep learning to facilitate the dynamic programming of deep networks according to their intended function. However, in the context of optimization, this idea is utilized to create the search space dynamically. This means that design-by-run enables us to build the search space based on the optimization goals, allowing for a more flexible and customized approach to optimization [45]. OPTUNA defines hyperparameter optimization as optimizing an objective function by selecting an appropriate set of hyperparameters to achieve the best possible validation score [46]. To achieve this, OPTUNA employs a trailing object to construct the objective function and dynamically creates the search space as the objective function runs using the trail object's methods. In other words, OPTUNA optimizes hyperparameters by iteratively refining the search-space based on the objective function's output.

#### 3.5.2. Sampling

Sampling is a crucial part of the optimization process in OPTUNA, a hyperparameter optimization framework. OPTUNA supports both independent sampling and relational sampling [38]. In independent sampling, each hyperparameter is sampled independently of other hyperparameters. OPTUNA has several separate sampling algorithms, such as random search, grid search, and the Tree-structured Parzen Estimator (TPE). Relational sampling, on the other hand, considers the correlations between hyperparameters. OPTUNA supports relational sampling through the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), a derivative-free optimization algorithm that can effectively explore high-dimensional and non-linear search spaces. The choice of sampling method in OPTUNA depends on the specific optimization task and the characteristics of the search space. Generally, independent sampling methods like TPE are more computationally efficient and work well for low-dimensional search spaces. In contrast, relational sampling methods like CMA-ES are better suited for high-dimensional search spaces with complex correlations between hyperparameters. OPTUNA also provides the flexibility to define custom sampling methods, allowing users to experiment with different sampling algorithms and tailor the optimization process to their needs.

#### 3.5.3. Pruning

Pruning is a technique used in hyperparameter optimization to reduce the computational resources needed for the optimization process [46]. OPTUNA provides several pruning algorithms to help users achieve faster and more efficient optimization. One of the most commonly used pruning algorithms in OPTUNA is successive halving, a type of early stopping technique [38]. In this method, the search space is split into multiple

groups of hyperparameters, and each group is evaluated in parallel. The worst-performing half of the groups are eliminated, and the process is repeated until only one group of hyperparameters remains [39]. This reduces the number of trials needed to find the optimal hyperparameters. OPTUNA also supports asynchronous successive halving, which allows trials to run in parallel and dynamically allocates more resources to promising trials while cutting resources for less promising ones. Overall, pruning algorithms in OPTUNA help to reduce the computational cost of hyperparameter optimization while improving the efficiency and effectiveness of the search for optimal hyperparameters.

### 3.5.4. Easy-to-Setup

By default, OPTUNA employs its memory data structure as a storage location; thus, making it effortless to use for simple purposes, which is an essential prerequisite for a contemporary hyperparameter optimization framework. OPTUNA offers numerous advanced capabilities, such as performing independent and relational sampling and providing various pruning algorithms, making it a superior framework for hyperparameter tuning. The architecture of OPTUNA is depicted in Figure 4. In OPTUNA, each worker in every sample executes the objective function (OF) once. The OF is executed through OPTUNA APIs, and when the moment API is called, it accesses the storage and retrieves any relevant data from previous samples stored in memory [39]. The workers operate independently, and they use the repository to track the results of the current study.

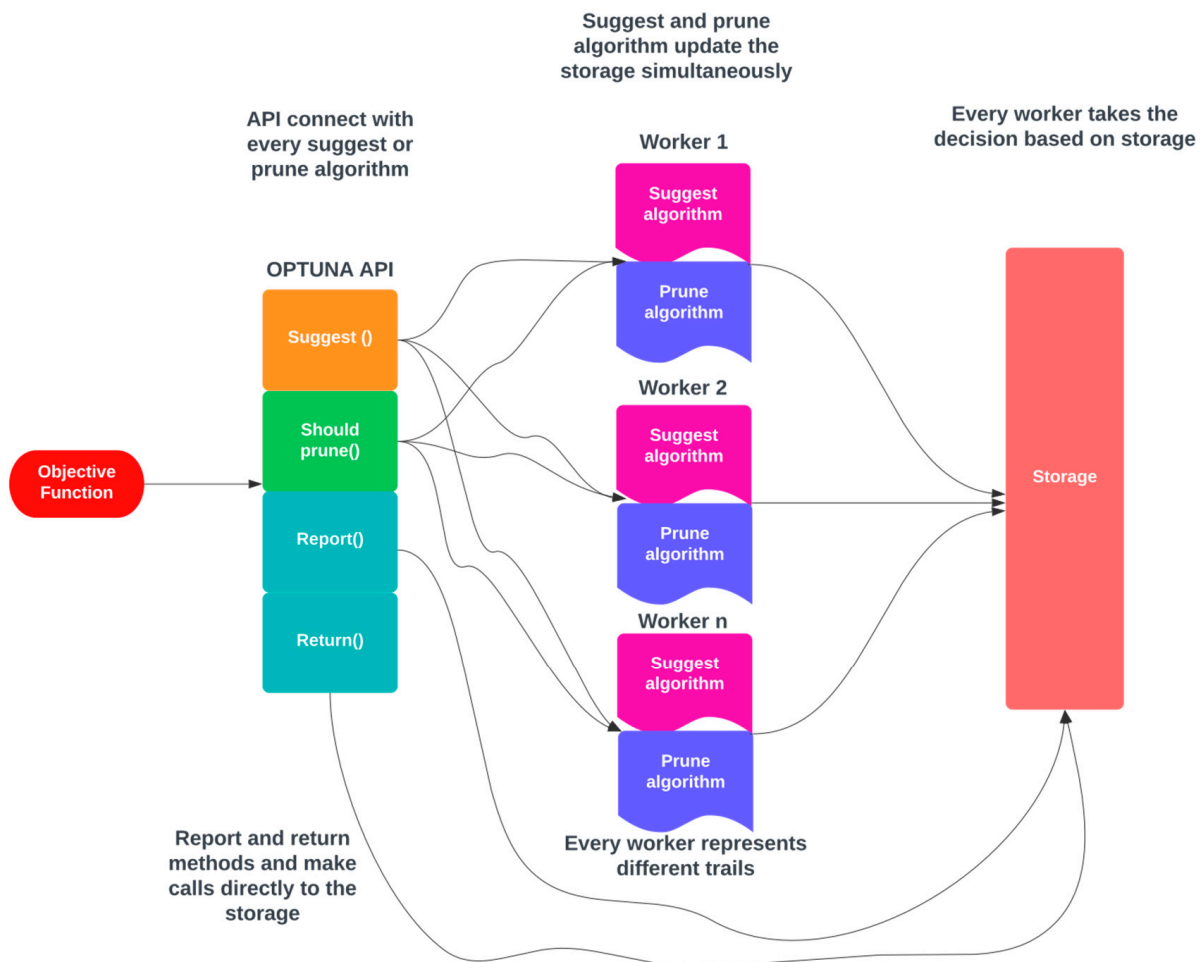


Figure 4. Architecture for OPTUNA.

### 3.6. Gradient Boosting

Gradient boosting (GB) is a commonly used machine learning approach for classification and regression tasks. It is an ensemble learning method that combines multiple weak learners (basic models) into a stronger one [47,48]. The basic idea of gradient boosting is to iteratively add new vulnerable learners to the model, with each new learner trained to correct the mistakes made by the previous ones [49]. Gradient boosting typically uses decision trees to make predictions [50]. The loss-function quantifies the difference between predicted and actual output and is frequently utilized as the objective function for gradient boosting classifiers. The most commonly used loss functions for classification problems are cross-entropy and exponential loss. The cross-entropy loss is defined as given in Equation (1):

$$L(y, f(x)) = - \sum_{i=1}^K y_i \log(f_i(x)) \tag{1}$$

where  $y_i \in [0, 1]$  is the true label for class  $i$ ,  $K$  is the number of classes, and  $f_i(x) \in [0, 1]$  is the predicted probability of class  $i$ . The exponential loss is defined as given in Equation (2):

$$L(y, f(x)) = \exp(-yf(x)) \tag{2}$$

where  $y \in [-1, 1]$  is the true label and  $f(x) \in \mathbb{R}$  is the predicted output of the model.

The objective function for gradient boosting is typically defined as the sum of the loss function over all the training examples, with the addition of a regularization term to prevent overfitting and given by Equation (3):

$$obj(\theta) = \sum_{i=1}^n L(y_i, \sum_{j=1}^m f_j(x_i, \theta)) + \sum_{j=1}^m \Omega(f_j) \tag{3}$$

where  $n$  is the number of training examples,  $m$  is the number of trees in the ensemble,  $f_j(x_i, \theta)$  is the output of the  $j$ -th tree for the  $i$ -th training example, and  $\Omega(f_j)$  is a regularization term that penalizes complex trees. The regularization term can take different forms, such as the L1 or L2 norm of the tree weights. The parameters  $\theta$  of the objective function are the parameters of the trees in the ensemble.

The gradient boosting classifier aims to minimize the objective function with respect to the tree parameters  $\theta$  using gradient descent. At each iteration, a new tree is added to the ensemble to reduce the residual error of the previous trees. The gradient of the objective function with respect to the model's predicted output is used to train the new tree. The mathematical algorithm for gradient boosting can be described in Algorithm 1:

---

**Algorithm 1:** Gradient Boosting Classifier

---

**Step 1:** Initialize the model by setting the initial predicted output of the model to be a constant value, such as the mean of the target variable.

**Step 2:** For  $m = 1$  to  $M$

- (a) Compute the negative gradient of the loss function with respect to the predicted output of the model for each training example:

$$r_{im} = - \left. \frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right|_{f(x)=f_{m-1}(x_i)}$$

where  $L(y_i, f(x_i))$  is the loss function,  $y_i$  is the true label for the  $i$ -th training example,  $f(x_i)$  is the predicted output of the model for the  $i$ -th training example, and  $f_{m-1}(x_i)$  is the predicted output of the model up to the  $m - 1$ -th iteration.

- (b) Fit a decision tree  $h_m(x)$  to the negative gradient values  $r_{im}$ :

$$h_m(x) = \operatorname{argmin}_h \sum_{i=1}^n [r_{im} - h(x_i)]^2$$

where  $h(x)$  is the output of the decision tree, and  $h_m(x)$  is the output of the  $m$ -th decision tree.

---

**Algorithm 1:** *Cont.*

- (c) Compute the step size
- $\gamma_m$
- by minimizing the following objective function

$$\gamma_m = \operatorname{argmin}_{\gamma} \sum_{i=1}^n L(y_i, f_{m-1}(x_i) + \gamma h_m(x_i))$$

where  $f_{m-1}(x_i)$  is the predicted output of the model up to the  $m - 1 - th$  iteration, and  $f_m(x_i) = f_{m-1}(x_i) + \gamma_m h_m(x_i)$  is the predicted output of the model after the  $m - th$  iteration.

- (d) Update the predicted output of the model:

$$f_m(x) = f_{m-1}(x) + \gamma_m h_m(x)$$

where  $f_{m-1}(x)$  is the predicted output of the model up to the  $m - 1 - th$  iteration,  $h_m(x)$  is the output of the  $m - th$  decision tree, and  $\gamma_m$  is the step size.

**Step 3:** Output the final predicted output of the model:

$$f_M(x) = \sum_{m=1}^M \gamma_m h_m(x)$$

where  $f_M(x)$  is the final predicted output of the model,  $M$  is the number of iterations,  $\gamma_m$  is the step size at the  $m - th$  iteration, and  $h_m(x)$  is the output of the  $m - th$  decision tree.

### 3.7. Proposed hyOPTGB Model

After Min-Max normalization and feature selection, we utilized the OPTUNA optimization to optimize the hyperparameters of the gradient boosting model. We opted for eight specific hyperparameters: `loss`, `learning_rate`, `n_estimators`, `subsample`, `criterion`, `min_samples_split`, `min_samples_leaf`, and `max_depth`. Following the tuning process, we trained the gradient boosting model using the optimized hyperparameters. The hyperparameter called `loss` is the loss function that will be optimized, which is employed for binomial and multinomial deviance. This function is especially beneficial for classification tasks that require probabilistic outputs. `Learning_rate` reduces the impact of each tree during the training process. `N_estimators` determine the number of boosting rounds to execute in gradient boosting. Gradient boosting is typically resistant to overfitting, so increasing the number of rounds often leads to improved performance. `Subsample` refers to the proportion of the total training samples to be utilized for training each base learner. `Criterion` refers to the metric used for evaluating the effectiveness of a split. `Min_samples_split` is a hyperparameter that sets the minimum number of data points that must be present in a decision tree node to permit a split. `Min_samples_leaf` is a hyperparameter that defines the minimum number of instances that should be present in a leaf node of a tree. The `Max_depth` hyperparameter sets the maximum depth a tree can reach by limiting the number of nodes. Tuning this parameter is crucial for achieving optimal performance, and the ideal value depends on how the input features interact. If set to "none", the tree will keep growing until all leaf nodes become pure or have fewer than the "min\_samples\_split" samples. Table 2 demonstrates the hyperparameters for the gradient boosting model using the default hyperparameters, and Table 3 illustrates the hyperparameters for the gradient boosting model using OPTUNA.

### 3.8. Machine Learning Classification Models

This section employs five different machine learning classification models, namely, decision tree (DT), support vector machine (SVM), dummy classifier (DC), ridge classifier (RC), and bagging classifier (BC), to evaluate and compare their performance with the proposed hyOPTGB model. Min-Max normalization and 15-fold cross-validation are performed for the five machine learning models using their default hyperparameters. The architecture for the five machine learning models is in Figure 5.

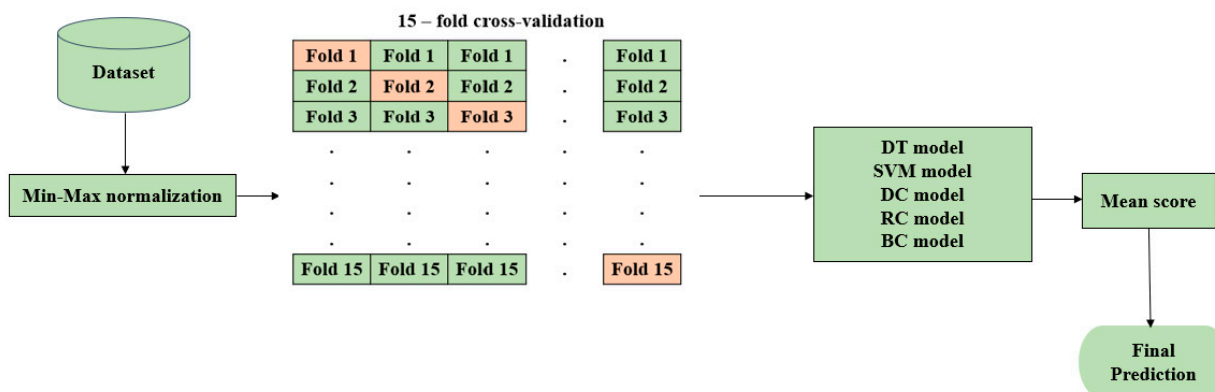


**Table 2.** Default hyperparameters for the gradient boosting model.

No.	Parameters	Default
1	Loss	Log_loss
2	Learning_rate	0.1
3	N_estimators	100
4	Subsample	1
5	Criterion	Friedman_mse
6	Min_samples_split	2
7	Min_samples_leaf	1
8	Max_depth	3

**Table 3.** Optimized hyperparameters for the gradient boosting model using OPTUNA.

No.	Parameters	Default
1	Loss	Deviance
2	Learning_rate	0.7
3	N_estimators	582
4	Subsample	0.4
5	Criterion	Squared_error
6	Min_samples_split	9
7	Min_samples_leaf	7
8	Max_depth	12



**Figure 5.** Architecture for the five machine learning classification models.

### 3.8.1. Decision Tree

A decision tree (DT) is a machine learning algorithm that builds a tree-like model of decisions and their possible consequences [51]. It is a popular approach for solving classification problems in which the input data is classified into multiple classes based on a set of rules derived from the training data. The decision tree algorithm recursively splits the data into subsets based on the features that best discriminate between the classes until a stopping criterion is met [52]. The resulting tree can classify new data by following the path from the root to a leaf node corresponding to the predicted class.

### 3.8.2. Support Vector Machine

Support vector machine (SVM) is a classification algorithm used in machine learning. It finds the optimal hyperplane that separates different classes in the input data [53]. In a two-class classification problem, SVM chooses the hyperplane that maximizes the margin between the two closest data points from different classes. The hyperplane is defined by the support vectors, which are the data points lying on the margin. SVMs can also handle non-linearly separable data by mapping it to a higher dimensional space using a kernel function [54]. This allows SVM to find the optimal hyperplane in the transformed space. SVMs are suitable for high-dimensional datasets and have widespread applications.

### 3.8.3. Dummy Classifier

A dummy classifier (DC) is a classification algorithm used as a baseline model to compare the performance of other more advanced classifiers [55]. It is a simple algorithm that makes predictions by following a pre-defined rule, such as always predicting the most frequent class in the training data or randomly selecting a class based on the class distribution. A dummy classifier is used to evaluate whether a more complex classifier can outperform this basic algorithm [56]. Dummy classifiers are helpful in situations where the class distribution is imbalanced or when the performance of a classifier needs to be compared against a simple baseline.

### 3.8.4. Ridge Classifier

A Ridge classifier (RC) is a linear classification algorithm that uses ridge regression to classify input data into multiple classes [57]. It works by finding the linear decision boundary that separates the different classes while minimizing the sum of the squared weights of the features. The regularization parameter, the ridge penalty, is added to the objective function to prevent overfitting of the model on the training data [58]. The ridge classifier is a variant of logistic regression that is particularly useful when dealing with high-dimensional datasets.

### 3.8.5. Bagging Classifier

A bagging classifier (BC) is an ensemble learning algorithm that combines multiple base classifiers to improve the overall classification performance [59]. It works by generating multiple bootstrap samples of the training data, and each sample is used to train a different base classifier. The base classifiers can be any type of classifier, such as decision trees or SVMs [60]. During the prediction phase, the bagging classifier aggregates the predictions of all base classifiers using a majority voting scheme to make the final prediction. Bagging reduces the model's variance and improves its generalization performance, especially when dealing with unstable base classifiers.

## 4. Results and Discussion

The Jupyter Notebook version 6.4.6 was utilized to carry out the experiments. It is a tool commonly employed for data analysis and visualization using Python. Jupyter Notebook offers a comprehensive environment to write, run, and document code and results, in addition to generating visual representations of data. This tool supports several programming languages, Python 3.8 included, and runs on a web browser. The experiments were executed on a computer that operates on Microsoft Windows 10, with an Intel Core i5 CPU and 16 GB of RAM. To assess the effectiveness of our models, we employed several metrics, including accuracy, recall, precision, and F1-score [61,62]. These metrics were used to evaluate and measure the model's performance in different aspects, such as correctly identifying true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \quad (4)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (5)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (6)$$

$$\text{F1-score} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}} \quad (7)$$

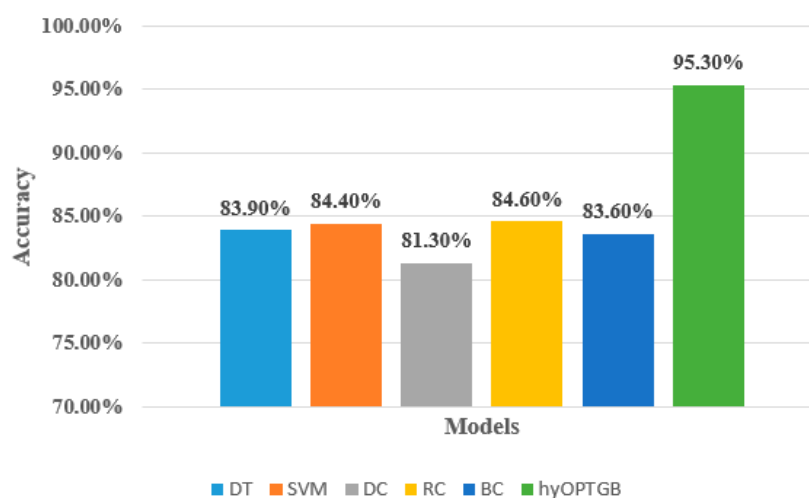
Our machine learning classifier for the model was the gradient boosting model. Initially, we optimized the hyperparameters of the gradient boosting model using the OP-TUNA framework. Once the hyperparameters were tuned, we used them to analyze the

model’s performance. OPTUNA used a distinct set of hyperparameters for the gradient boosting model during the tuning phase, which helped enhance performance on 15-fold cross-validation. Also, Min-Max normalization and forward selection were applied to the dataset. These features selected by forward selection include BMI, fever, diarrhea, epigastric pain, WBC, HGB, plat, ALT 1, ALT 4, ALT 24, base RNA, RNA 4, and RNA EF. We assessed and contrasted the performance of five machine learning classification models, specifically the decision tree (DT), support vector machine (SVM), dummy classifier (DC), ridge classifier (RC), and bagging classifier (BC), against the proposed hyOPTGB model. We used Min-Max normalization and 15-fold cross-validation for the five machine learning models to conduct this evaluation using their default hyperparameters. Table 4 displays the accuracy, F1-score, recall, and precision results of the five machine learning models and the proposed hyOPTGB model for performance evaluation.

**Table 4.** Performance of the five classification models and the proposed hyOPTGB model.

Models	Accuracy	F1-Score	Recall	Precision
DT	83.9%	83.5%	83.2%	82.9%
SVM	84.4%	84.1%	83.5%	83.2%
DC	81.3%	80.7%	80.4%	80.2%
RC	84.6%	84.4%	83.7%	83.3%
BC	83.6%	83.2%	82.8%	82.5%
hyOPTGB	95.3%	94.8%	94.5%	94.1%

As seen in Table 4, the hyOPTGB model outperformed all other models, achieving an accuracy of 95.3%, F1-score of 94.8%, recall of 94.5%, and precision of 94.1%. The RC model comes in second place, with high metrics scores. Its accuracy, F1-score, recall, and precision are 84.6%, 84.4%, 83.7%, and 83.3%, respectively. The SVM model comes in third place; its accuracy, F1-score, recall, and precision are 84.4%, 84.1%, 83.5%, and 83.2%, respectively. The DT model comes in fourth place; its accuracy, F1-score, recall, and precision are 83.9%, 83.5%, 83.2%, and 82.9%, respectively. The accuracy, F1-score, recall, and precision are 83.6%, 83.2%, 82.8%, and 82.5%, respectively, for the BC model, which comes in fifth place. The DC model obtains the worst results; its accuracy, F1-score, recall, and precision are 81.3%, 80.7%, 80.4%, and 80.2%, respectively. Figure 6 displays the accuracy of the five classification models and the proposed hyOPTGB model.



**Figure 6.** Comparison between the proposed hyOPTGB model and different machine learning models in terms of accuracy.

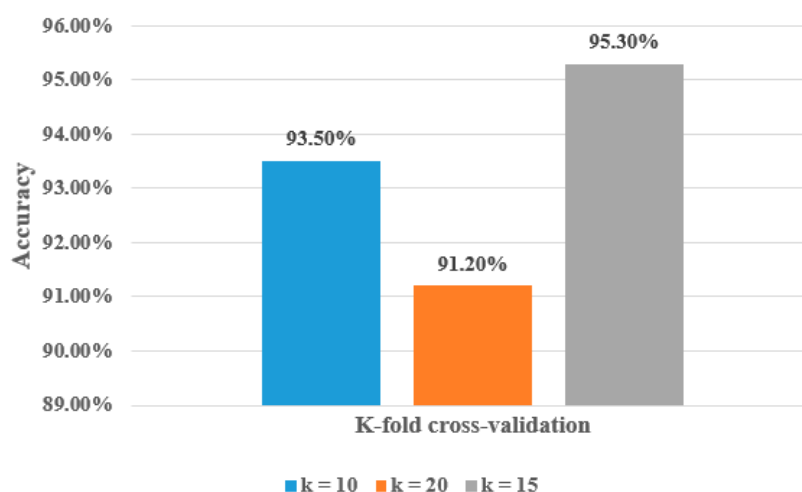
To ensure the strength and validity of the proposed model, we applied 10-fold cross-validation and 20-fold cross-validation on the hyOPTGB model and compared the results when using 15-fold cross-validation. Table 5 demonstrates the performance of the proposed

hyOPTGB model using 10-fold cross-validation, 15-fold cross-validation, and 20-fold cross-validation, respectively.

**Table 5.** Performance of the proposed hyOPTGB model using different k-fold cross-validation.

K-Fold Cross-Validation	Accuracy	F1-Score	Recall	Precision
k = 10	93.5%	93.1%	92.8%	92.7%
k = 15	<b>95.3%</b>	<b>94.8%</b>	<b>94.5%</b>	<b>94.1%</b>
k = 20	91.2%	90.9%	90.6%	90.2%

As seen in Table 5, the best results are obtained when (k = 15); its accuracy, F1-score, recall, and precision are 95.3%, 94.8%, 94.5%, and 94.1%, respectively. The worst results are obtained when (k = 20); its accuracy, F1-score, recall, and precision are 91.2%, 90.9%, 90.6%, and 90.2%, respectively. The accuracy, F1-score, recall, and precision are 93.5%, 93.1%, 92.8%, and 92.7%, respectively, when (k = 10). Figure 7 demonstrates the accuracy percentage for the proposed hyOPTGB model using 10-fold cross-validation, 15-fold cross-validation, and 20-fold cross-validation, respectively. Table 6 displays a comparative study that used the same dataset in this paper.



**Figure 7.** Accuracy percentage for the proposed hyOPTGB model using 10-fold cross-validation, 15-fold cross-validation, and 20-fold cross-validation in terms of accuracy.

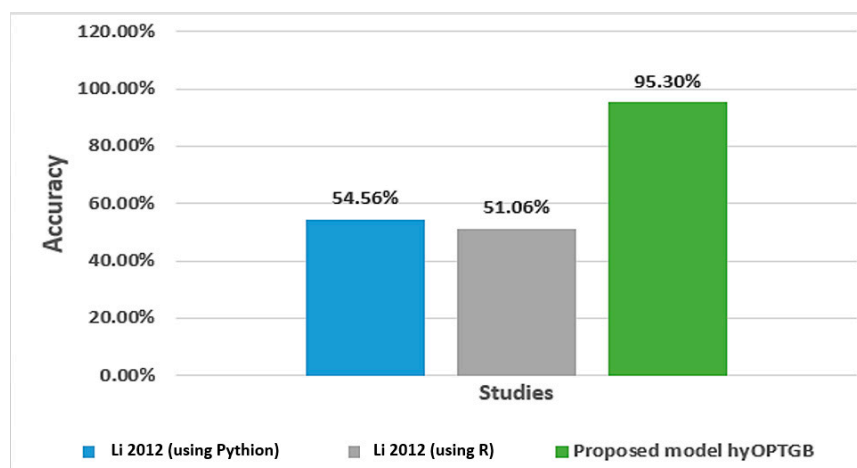
**Table 6.** Comparative study of this work with another study used the same dataset.

Studies	Model	Accuracy
Ref. [27]	Random Forest Using Python	54.56%
Ref. [27]	KNN Using R	51.06%
Proposed model hyOPTGB	Gradient boosting using OPTUNA	<b>95.3%</b>

As shown in Table 6, the proposed hyOPTGB model achieved better performance in terms of accuracy than the previous study. Figure 8 displays the accuracy percentage for the proposed hyOPTGB model, and previous studies used the same dataset.

When evaluating the results of the hyOPTGB model, accuracy is the criterion that is applied. Table 7 presents the results of twenty separate iterations of the hyOPTGB model. There is information supplied regarding these runs’ minimal, median, maximum, and mean accuracy. These data, taken from a large number of different model runs, enable an evaluation of the consistency and efficiency of the hyOPTGB model. Because of this explanation’s level of depth, you are able to evaluate the performance and dependability of the model.





**Figure 8.** Accuracy percentage for the proposed hyOPTGB model, and previous studies used the same dataset [27].

**Table 7.** A description of the hyOPTGB model that was proposed as well as the results of other models based on the accuracy factor.

	Random Forest	KNN	MLP	LSTM	hyOPTGB
Number of values	20	20	20	20	20
Minimum	0.5256	0.5006	0.7489	0.8976	0.953
25% Percentile	0.5456	0.5106	0.7589	0.9076	0.953
Median	0.5456	0.5106	0.7589	0.9076	0.953
75% Percentile	0.5456	0.5106	0.7589	0.9076	0.953
Maximum	0.5756	0.5306	0.7989	0.9276	0.973
Range	0.05	0.03	0.05	0.03	0.02
Mean	0.5468	0.5116	0.7614	0.9086	0.955
Std. Deviation	0.01012	0.005525	0.0102	0.005525	0.005231
Std. Error of Mean	0.002264	0.001235	0.00228	0.001235	0.00117
Sum	10.94	10.23	15.23	18.17	19.1

The results of the hyOPTGB and comparison model ANOVA are presented in Table 8. The purpose of this statistical study is to investigate and explain model differences. The results of an ANOVA can demonstrate whether or not there is statistical variation in model performance. The Wilcoxon signed-rank test is used to make a comparison between the hyOPTGB model and the models that are being tested in Table 9. The results of this non-parametric test are compared based on matching data, such as how well a model performed on the same dataset. The Wilcoxon signed-rank test as well as ten distinct iterations of each model make it possible to make accurate comparisons, which in turn increases the reliability of the study. The hyOPTGB model is evaluated in an objective manner using these statistical tests, in comparison to the other models. The analysis of variance (ANOVA) and the Wilcoxon signed-rank test (Wilcoxon *p*-values) are both useful tools for determining whether or not differences in model performance are statistically significant. These findings help explain why the hyOPTGB model is more effective than other models and why it can be applied to this particular job or dataset.

**Table 8.** The ANOVA test for the presented hyOPTGB model and other models.

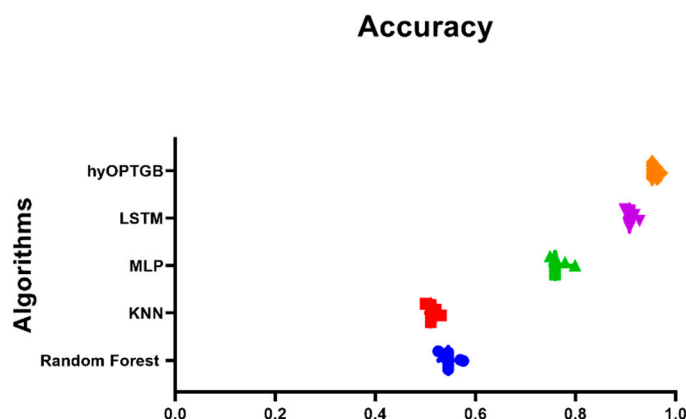
	SS	DF	MS	F (DFn, DFd)	<i>p</i> Value
Treatment (between columns)	3.291	4	0.8227	F (4, 95) = 13,952	<i>p</i> < 0.0001
Residual (within columns)	0.005602	95	$5.9 \times 10^{-5}$	-	-
Total	3.297	99	-	-	-

**Table 9.** The Wilcoxon signed-rank test for the presented hyOPTGB model and other models.

	Random Forest	KNN	MLP	LSTM	hyOPTGB
Theoretical median	0	0	0	0	0
Actual median	0.5456	0.5106	0.7589	0.9076	0.953
Number of values	20	20	20	20	20
Wilcoxon signed-rank test					
Sum of signed ranks (W)	210	210	210	210	210
Sum of positive ranks	210	210	210	210	210
Sum of negative ranks	0	0	0	0	0
<i>p</i> value (two tailed)	<0.0001	<0.0001	<0.0001	<0.0001	<0.0001
Exact or estimate?	Exact	Exact	Exact	Exact	Exact
Significant (alpha = 0.05)?	Yes	Yes	Yes	Yes	Yes
How big is the discrepancy?					
Discrepancy	0.5456	0.5106	0.7589	0.9076	0.953

The comparison of the hyOPTGB model with other accuracy-based models in Figure 9 provides a valuable visual representation of each model’s performance in terms of accuracy. This comparison can offer several insights and benefits:

1. **Model Performance Assessment:** The plot allows readers to quickly assess the performance of various models in a single view. This is particularly important when evaluating machine learning models, as accuracy is a common metric for measuring predictive performance.
2. **Model Ranking:** By plotting the accuracy achieved by each model, it becomes apparent which model outperforms the others and to what extent. This ranking can help identify the most effective model for the specific task or dataset under consideration.
3. **Identification of the Best Model:** It aids in the identification of the best-performing model, which can be crucial for decision-making in practical applications.
4. **Comparative Analysis:** The plot enables a side-by-side comparison of the hyOPTGB model with other models, hyOPTGB model excels in terms of accuracy.
5. **Validation of Results:** Plots of model accuracy provide visual evidence of the research findings, making it easier for readers and reviewers to validate the reported results and conclusions.



**Figure 9.** Plot of the presented hyOPTGB model and other models based on the accuracy.

The histograms of accuracy values displayed in Figure 10 provide a valuable visual representation of the dispersion and concentration of the model performance results, both for the hyOPTGB model and the comparative models. Histograms are effective tools for understanding the distribution of accuracy values. They reveal how the performance of each model is dispersed across different accuracy levels. This can help identify the range and variability of the results.

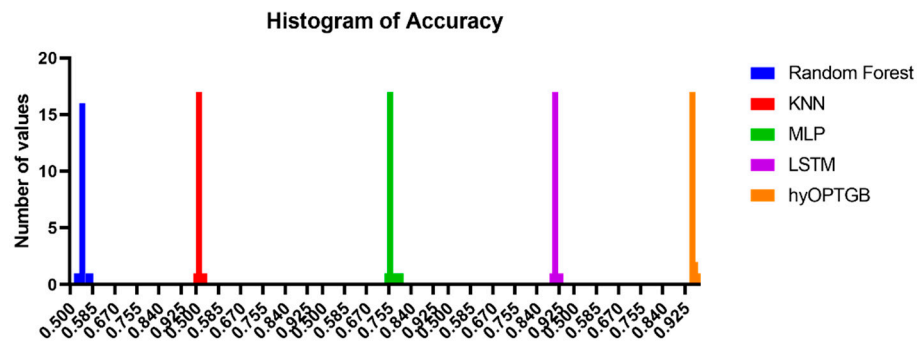


Figure 10. Histogram of accuracy for the presented hyOPTGB model and other models.

The inclusion of residual plots, homoscedasticity plots, QQ (Quantile–Quantile) plots, and heat maps for the hyOPTGB model and comparative models in Figure 11 offers a comprehensive assessment of the model performance and various aspects of the models. Here’s why these visualizations are significant:

1. Residual Analysis: Residual plots allow for the examination of the differences between observed values and model predictions. They help identify patterns or deviations in the model’s errors, such as systematic biases or outliers.

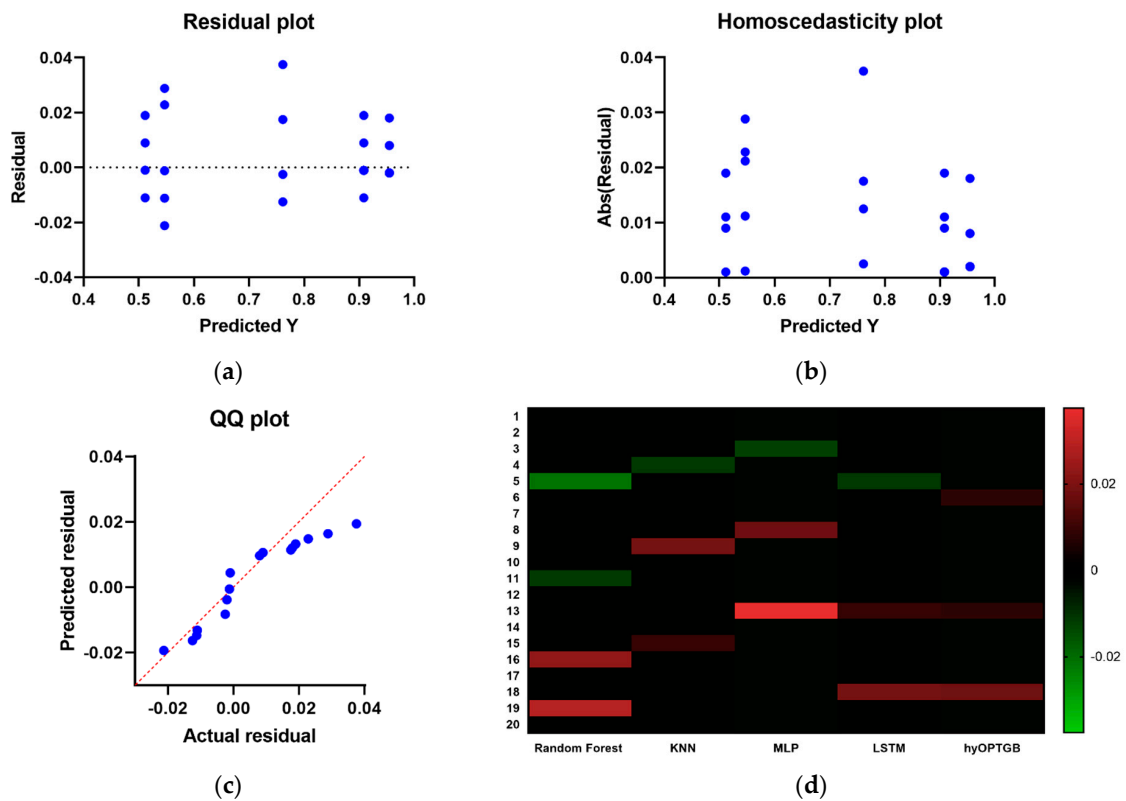


Figure 11. The residual plots, homoscedasticity plots, QQ plots and the heat maps used to compare and display the presented hyOPTGB model to other models.

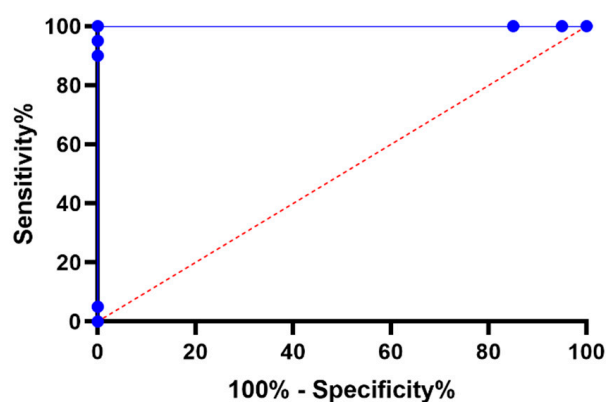
2. Homoscedasticity Assessment: Homoscedasticity plots are used to check the constant variance of model residuals across different levels of the predictor variable. They help ensure that the model’s errors have consistent variance and do not exhibit a trend.
3. QQ Plots for Normality: QQ plots are particularly useful for assessing whether the residuals follow a normal distribution. Deviations from the expected straight line in a QQ plot can indicate departures from normality in the residuals.
4. Heat Maps for Correlation: Heat maps provide a visual representation of the correlation between variables. In the context of machine learning, they can illustrate

the correlation between features, residuals, or model predictions, helping to identify multicollinearity or patterns in the data.

The comparison of ROC curves presented in Figure 12 is a crucial element in evaluating and understanding the performance of binary classification models, specifically the hyOPTGB model and the LSTM model. Here's why these ROC curves are significant:

1. **Assessment of Binary Classification:** ROC curves are a standard tool for assessing the performance of binary classification models. They help to understand how well a model can discriminate between the positive class and the negative class by varying the classification threshold.
2. **Sensitivity and Specificity Trade-off:** The ROC curve graphically depicts the trade-off between sensitivity (the true positive rate) and specificity (the true negative rate) as the threshold for classification changes. It shows how different decision criteria can affect the balance between correctly identifying positive cases (sensitivity) and correctly identifying negative cases (specificity).
3. **Model Comparison:** By comparing the ROC curves of two different models, such as the hyOPTGB model and the LSTM model, one can assess and visually contrast their classification performance. This comparison can provide insights into which model performs better at different operating points and threshold settings.
4. **Area Under the Curve (AUC):** The area under the ROC curve (AUC) is often used as a single summary measure of a model's overall performance. A higher AUC indicates a better model, as it represents a larger area under the ROC curve and suggests a better balance between sensitivity and specificity.

#### ROC curve (hyOPTGB : LSTM )



**Figure 12.** The ROC curve of the presented hyOPTGB model and LSTM model.

#### 5. Discussion

The novelty of the study can be summarized in several key aspects:

1. **Targeted Research Area:** The paper focuses on hepatitis C virus (HCV) infection in Egypt, a region with one of the highest HCV prevalence rates globally. This specific geographic context is significant, as it addresses a pressing health issue in a region with distinct risk factors and challenges.
2. **Unique Dataset:** The study employs a dataset containing 1385 instances and 29 features related to HCV in Egypt, sourced from the UCI machine learning repository. This dataset is a valuable resource and forms the basis of the research, contributing to the understanding of HCV in this context.
3. **Development of hyOPTGB Model:** A novel predictive model called hyOPTGB, which leverages an optimized gradient boosting (GB) classifier. The novelty here lies in the development of this specific model tailored to predict HCV disease in Egypt. The optimization of hyperparameters with the OPTUNA framework further enhances the model's performance.



4. **Feature Selection and Preprocessing:** The paper applies Min-Max normalization as a preprocessing step for scaling the dataset values and utilizes the forward selection (FS) wrapped method for identifying essential features. These techniques are integral to the model's performance and contribute to the overall methodology's uniqueness.
5. **Comparative Analysis:** This paper conducts a comprehensive comparison of the hyOPTGB model against five other machine learning models, including decision tree (DT), support vector machine (SVM), dummy classifier (DC), ridge classifier (RC), and bagging classifier (BC). This comparative analysis helps demonstrate the superiority of their proposed model.
6. **High Accuracy Rate:** One of the key findings is that the hyOPTGB model outperformed the other machine learning models with a remarkable 95.3% accuracy rate. Such a high accuracy rate in the prediction of HCV in Egypt is a notable contribution, indicating the efficacy of their model.
7. **Comparison with Other Studies:** The paper also compares the hyOPTGB model with models proposed by different authors who used the same dataset. This comparative analysis helps establish the superiority of their model within the context of existing research.

## 6. Conclusions and Future Work

This research paper presents hyOPTGB, a model that employs an optimized gradient boosting (GB) classifier to predict HCV disease in Egypt. The model's accuracy is enhanced using hyperparameter tuning with the OPTUNA framework, while the essential features in the dataset are identified using the forward selection (FS) wrapped method. Also, the Min-Max normalization preprocessing technique is used to scale the values of a dataset to a fixed range. The UCI machine learning repository provided the dataset containing 1385 instances and 29 features. The study compares hyOPTGB with five other machine learning models, namely, decision tree (DT), support vector machine (SVM), dummy classifier (DC), ridge classifier (RC), and bagging classifier (BC). It evaluates their efficiency using accuracy, recall, precision, and F1-score. The five machine learning models used their default hyperparameters. The hyOPTGB model outperforms the other machine learning models, achieving a 95.3% accuracy. The paper also conducts a comparative study of the proposed hyOPTGB model against those used by other researchers who employed the same dataset, and the results depicted that the proposed model achieved the best results. Some potential future directions for HCV disease prediction can be conducted in the future as (1) using genetic information; this can inform the development of models that incorporate genetic information to predict HCV infection risk. This approach can help identify genetic markers associated with HCV susceptibility, which can inform targeted prevention and intervention efforts. (2) Integrating multiple data sources by combining data from multiple sources, such as electronic health records, public health surveillance data, and social media, can improve the accuracy of predictive models and identify novel risk factors for HCV infection.

**Author Contributions:** Methodology, S.M.T.; Software, A.M.E.; Validation, A.H.A.; Formal analysis, A.I.; Investigation, N.K.; Resources, A.A.A.; Data curation, M.M.E.; Writing—original draft, M.Y.S.; Writing—review & editing, D.S.K. and Z.T.; Visualization, L.A. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R120), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data is available up on request.

**Acknowledgments:** Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R120), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

- Robertson, B.; Myers, G.; Howard, C.; Brettin, T.; Bukh, J.; Gaschen, B.; Gojobori, T.; Maertens, G.; Mizokami, M.; Nainan, O.; et al. Classification, nomenclature, and database development for hepatitis C virus (HCV) and related viruses: Proposals for standardization. *Arch. Virol.* **1998**, *143*, 2493–2503. [[CrossRef](#)] [[PubMed](#)]
- Smith, D.B.; Bukh, J.; Kuiken, C.; Muerhoff, A.S.; Rice, C.M.; Stapleton, J.T.; Simmonds, P. Expanded classification of hepatitis C virus into 7 genotypes and 67 subtypes: Updated criteria and genotype assignment web resource. *Hepatology* **2014**, *59*, 318–327. [[CrossRef](#)] [[PubMed](#)]
- Borgia, S.M.; Hedskog, C.; Parhy, B.; Hyland, R.H.; Stamm, L.M.; Brainard, D.M.; Subramanian, M.G.; McHutchison, J.G.; Mo, H.; Svarovskaia, E.; et al. Identification of a novel hepatitis C virus genotype from Punjab, India: Expanding classification of hepatitis C virus into 8 genotypes. *J. Infect. Dis.* **2018**, *218*, 1722–1729. [[CrossRef](#)] [[PubMed](#)]
- Louie, K.S.; St Laurent, S.; Forssen, U.M.; Mundy, L.M.; Pimenta, J.M. The high comorbidity burden of the hepatitis C virus infected population in the United States. *BMC Infect. Dis.* **2012**, *12*, 86. [[CrossRef](#)]
- Simmonds, P.; Bukh, J.; Combet, C.; Deléage, G.; Enomoto, N.; Feinstone, S.; Halfon, P.; Inchauspé, G.; Kuiken, C.; Maertens, G.; et al. Consensus proposals for a unified system of nomenclature of hepatitis C virus genotypes. *Hepatology* **2005**, *42*, 962–973. [[CrossRef](#)]
- Huang, T.; Wang, J.; Cai, Y.D.; Yu, H.; Chou, K.C. Hepatitis C virus network based classification of hepatocellular cirrhosis and carcinoma. *PLoS ONE* **2012**, *7*, e34460. [[CrossRef](#)]
- Mohamed, A.A.; Elbedewy, T.A.; El-Serafy, M.; El-Toukhy, N.; Ahmed, W.; El Din, Z.A. Hepatitis C virus: A global view. *World J. Hepatol.* **2015**, *7*, 2676–2680. [[CrossRef](#)]
- Huang, R.; Rao, H.; Yang, M.; Gao, Y.; Wang, J.; Jin, Q.; Ma, D.; Wei, L. Noninvasive measurements predict liver fibrosis well in hepatitis C virus patients after direct-acting antiviral therapy. *Dig. Dis. Sci.* **2020**, *65*, 1491–1500. [[CrossRef](#)]
- Westermann, C.; Peters, C.; Lisiak, B.; Lamberti, M.; Nienhaus, A. The prevalence of hepatitis C among healthcare workers: A systematic review and meta-analysis. *Occup. Environ. Med.* **2015**, *72*, 880–888. [[CrossRef](#)]
- Yağanoğlu, M. Hepatitis C virus data analysis and prediction using machine learning. *Data Knowl. Eng.* **2022**, *142*, 102087. [[CrossRef](#)]
- Kashif, A.A.; Bakhtawar, B.; Akhtar, A.; Akhtar, S.; Aziz, N.; Javeid, M.S. Treatment response prediction in hepatitis C patients using machine learning techniques. *Int. J. Technol. Innov. Manag.* **2021**, *1*, 79–89. [[CrossRef](#)]
- Schratz, P.; Muenchow, J.; Iturrutxa, E.; Richter, J.; Brenning, A. Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data. *Ecol. Model.* **2019**, *406*, 109–120. [[CrossRef](#)]
- Yang, L.; Shami, A. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing* **2020**, *415*, 295–316. [[CrossRef](#)]
- Abdelhamid, A.A.; El-Kenawy, E.S.M.; Khodadadi, N.; Mirjalili, S.; Khafaga, D.S.; Alharbi, A.H.; Ibrahim, A.; Eid, M.M.; Saber, M. Classification of monkeypox images based on transfer learning and the al-biruni earth radius optimization algorithm. *Mathematics* **2022**, *10*, 3614. [[CrossRef](#)]
- Eid, M.M.; El-Kenawy, E.S.M.; Khodadadi, N.; Mirjalili, S.; Khodadadi, E.; Abotaleb, M.; Alharbi, A.H.; Abdelhamid, A.A.; Ibrahim, A.; Amer, G.M.; et al. Meta-heuristic optimization of LSTM-based deep network for boosting the prediction of monkeypox cases. *Mathematics* **2022**, *10*, 3845. [[CrossRef](#)]
- Khafaga, D.S.; Ibrahim, A.; El-Kenawy, E.S.M.; Abdelhamid, A.A.; Karim, F.K.; Mirjalili, S.; Khodadadi, N.; Lim, W.H.; Eid, M.M.; Ghoneim, M.E. An Al-Biruni earth radius optimization-based deep convolutional neural network for classifying monkeypox disease. *Diagnostics* **2022**, *12*, 2892. [[CrossRef](#)]
- Khafaga, D. Meta-heuristics for feature selection and classification in diagnostic breast cancer. *Comput. Mater. Contin.* **2022**, *73*, 749–765.
- Alhussan, A.A.; Abdelhamid, A.A.; Towfek, S.K.; Ibrahim, A.; Eid, M.M.; Khafaga, D.S.; Saraya, M.S. Classification of diabetes using feature selection and hybrid Al-Biruni earth radius and dipper throated optimization. *Diagnostics* **2023**, *13*, 2038. [[CrossRef](#)]
- Samee, N.A.; El-Kenawy, E.S.M.; Atteia, G.; Jamjoom, M.M.; Ibrahim, A.; Abdelhamid, A.A.; El-Attar, N.E.; Gaber, T.; Slowik, A.; Shams, M.Y. Metaheuristic optimization through deep learning classification of COVID-19 in chest X-ray images. *Comput. Mater. Contin.* **2022**, *73*, 4193–4210.
- Sanaa, K.; ElEleimy, M.; Hegazy, D.; Nasr, M. Hepatitis C Virus (HCV) for Egyptian patients. *UCI Mach. Learn. Repos.* **2019**. [[CrossRef](#)]
- Cai, J.; Luo, J.; Wang, S.; Yang, S. Feature selection in machine learning: A new perspective. *Neurocomputing* **2018**, *300*, 70–79. [[CrossRef](#)]
- Kotsiantis, S. Feature selection for machine learning classification problems: A recent overview. *Artif. Intell. Rev.* **2011**, *42*, 157–176. [[CrossRef](#)]

23. Salcedo-Sanz, S.; Cornejo-Bueno, L.; Prieto, L.; Paredes, D.; García-Herrera, R. Feature selection in machine learning prediction systems for renewable energy applications. *Renew. Sustain. Energy Rev.* **2018**, *90*, 728–741. [[CrossRef](#)]
24. Tsvetkov, V.; Tokin, I.; Lioznov, D. Machine learning model for diagnosing the stage of liver fibrosis in patients with chronic viral hepatitis C. *Preprints* **2021**, 2021020488. [[CrossRef](#)]
25. Akella, A.; Akella, S. Applying machine learning to evaluate for fibrosis in chronic hepatitis C. *medRxiv* **2020**. [[CrossRef](#)]
26. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Ann. Stat.* **2001**, *29*, 1189–1232. [[CrossRef](#)]
27. Li, P. Robust logitboost and adaptive base class (abc) logitboost. *arXiv* **2012**, arXiv:1203.3491.
28. Tran, N.; Schneider, J.G.; Weber, I.; Qin, A.K. Hyper-parameter optimization in classification: To-do or not-to-do. *Pattern Recognit.* **2020**, *103*, 107245. [[CrossRef](#)]
29. Nugroho, A.; Suhartanto, H. September. Hyper-parameter tuning based on random search for densenet optimization. In Proceedings of the 2020 7th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), Semarang, Indonesia, 24–25 September 2020; pp. 96–99.
30. Cai, Z.; Long, Y.; Shao, L. Classification complexity assessment for hyper-parameter optimization. *Pattern Recognit. Lett.* **2019**, *125*, 396–403. [[CrossRef](#)]
31. Nandipati, S.C.; XinYing, C.; Wah, K.K. Hepatitis C virus (HCV) prediction by machine learning techniques. *Appl. Model. Simul.* **2020**, *4*, 89–100.
32. Abd El-Salam, S.M.; Ezz, M.M.; Hashem, S.; Elakel, W.; Salama, R.; ElMakhzangy, H.; ElHefnawi, M. Performance of machine learning approaches on prediction of esophageal varices for Egyptian chronic hepatitis C patients. *Inform. Med. Unlocked* **2019**, *17*, 100267. [[CrossRef](#)]
33. Hashem, S.; ElHefnawi, M.; Habashy, S.; El-Adawy, M.; Esmat, G.; Elakel, W.; Abdelazziz, A.O.; Nabeel, M.M.; Abdelmaksoud, A.H.; Elbaz, T.M.; et al. Machine learning prediction models for diagnosing hepatocellular carcinoma with HCV-related chronic liver disease. *Comput. Methods Programs Biomed.* **2020**, *196*, 105551. [[CrossRef](#)]
34. Sartakhti, J.S.; Zangoeei, M.H.; Mozafari, K. Hepatitis disease diagnosis using a novel hybrid method based on support vector machine and simulated annealing (SVM-SA). *Comput. Methods Programs Biomed.* **2012**, *108*, 570–579. [[CrossRef](#)] [[PubMed](#)]
35. Vikas, B.; Yaswanth, D.V.S.; Vinay, W.; Sridhar Reddy, B.; Saranyu, A.V.H. Classification of Hepatitis C Virus Using Case-Based Reasoning (CBR) with Correlation Lift Metric. In Proceedings of the Information Systems Design and Intelligent Applications: Proceedings of Fourth International Conference INDIA 2017, Birmingham, UK, 6–8 September 2018; Springer: Singapore, 2018; pp. 916–923.
36. Zaki, A.; Salama, M.A.; Hefny, H.; Hassani, A.E. Rough sets-based rules generation approach: A hepatitis c virus data sets. In Proceedings of the Advanced Machine Learning Technologies and Applications: First International Conference, AMLTA 2012, Cairo, Egypt, 8–10 December 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 52–59.
37. Kayvan Joo, A.H.; Ebrahimi, M.; Haqshenas, G. Prediction of hepatitis C virus interferon/ribavirin therapy outcome based on viral nucleotide attributes using machine learning algorithms. *BMC Res. Notes* **2014**, *7*, 565.
38. Lai, J.P.; Lin, Y.L.; Lin, H.C.; Shih, C.Y.; Wang, Y.P.; Pai, P.F. Tree-Based Machine Learning Models with Optuna in Predicting Impedance Values for Circuit Analysis. *Micromachines* **2023**, *14*, 265. [[CrossRef](#)]
39. Tonmoy, S.T.I.; Zaman, S.M. OOG-Optuna Optimized GAN Sampling Technique for Tabular Imbalanced Malware Data. In Proceedings of the 2022 IEEE International Conference on Big Data (Big Data), Osaka, Japan, 17–20 December 2022; pp. 6534–6539.
40. Elshewey, A.M.; Shams, M.Y.; El-Rashidy, N.; Elhady, A.M.; Shohieb, S.M.; Tarek, Z. Bayesian Optimization with Support Vector Machine Model for Parkinson Disease Classification. *Sensors* **2023**, *23*, 2085. [[CrossRef](#)]
41. Meyer, H.; Reudenbach, C.; Hengl, T.; Katurji, M.; Nauss, T. Improving performance of spatio-temporal machine learning models using forward feature selection and target-oriented validation. *Environ. Model. Softw.* **2018**, *101*, 1–9. [[CrossRef](#)]
42. Dietterich, T.G. Machine learning for sequential data: A review. In Proceedings of the Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshops SSPR 2002 and SPR 2002, Windsor, ON, Canada, 6–9 August 2002; Springer: Berlin/Heidelberg, Germany, 2002; pp. 15–30.
43. Alkhamash, E.H.; Hadjouni, M.; Elshewey, A.M. A Hybrid Ensemble Stacking Model for Gender Voice Recognition Approach. *Electronics* **2022**, *11*, 1750. [[CrossRef](#)]
44. Wong, T.T.; Yeh, P.Y. Reliable accuracy estimates from k-fold cross validation. *IEEE Trans. Knowl. Data Eng.* **2019**, *32*, 1586–1594. [[CrossRef](#)]
45. Agrawal, T.; Agrawal, T. Optuna and autoML. In *Hyperparameter Optimization in Machine Learning: Make Your Machine Learning and Deep Learning Models More Efficient*; Apress: New York, NY, USA, 2021; pp. 109–129.
46. Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; Koyama, M. Optuna: A next-generation hyperparameter optimization framework. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 2623–2631.
47. Peter, S.; Diego, F.; Hamprecht, F.A.; Nadler, B. Cost efficient gradient boosting. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–11.
48. Lusa, L. Gradient boosting for high-dimensional prediction of rare events. *Comput. Stat. Data Anal.* **2017**, *113*, 19–37.
49. Son, J.; Jung, I.; Park, K.; Han, B. Tracking-by-segmentation with online gradient boosting decision tree. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 3056–3064.

50. Chakrabarty, N.; Kundu, T.; Dandapat, S.; Sarkar, A.; Kole, D.K. Flight arrival delay prediction using gradient boosting classifier. In *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2018, Volume 2*; Springer: Singapore, 2019; pp. 651–659.
51. Rastogi, R.; Shim, K. PUBLIC: A decision tree classifier that integrates building and pruning. *Data Min. Knowl. Discov.* **2000**, *4*, 315–344. [[CrossRef](#)]
52. Priyanka; Kumar, D. Decision tree classifier: A detailed survey. *Int. J. Inf. Decis. Sci.* **2020**, *12*, 246–269.
53. Cervantes, J.; Garcia-Lamont, F.; Rodríguez-Mazahua, L.; Lopez, A. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing* **2020**, *408*, 189–215. [[CrossRef](#)]
54. Shams, M.Y.; El-kenawy, E.M.; Ibrahim, A.; Elshewey, A.M. A Hybrid Dipper Throated Optimization Algorithm and Particle Swarm Optimization (DTPSO) Model for Hepatocellular Carcinoma (HCC) Prediction. *Biomed. Signal Process. Control* **2023**, *85*, 104908. [[CrossRef](#)]
55. Martino, A.; Rizzi, A.; Mascioli, F.M.F. Supervised approaches for protein function prediction by topological data analysis. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
56. Liu, Z.; Chen, G.; Li, Z.; Kang, Y.; Qu, S.; Jiang, C. PSDC: A Prototype-Based Shared-Dummy Classifier Model for Open-Set Domain Adaptation. *IEEE Trans. Cybern.* **2022**, *53*, 7353–7366. [[CrossRef](#)]
57. Singh, A.; Prakash, B.S.; Chandrasekaran, K. A comparison of linear discriminant analysis and ridge classifier on Twitter data. In Proceedings of the 2016 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, India, 29–30 April 2016; pp. 133–138.
58. Peng, C.; Cheng, Q. Discriminative ridge machine: A classifier for high-dimensional data or imbalanced data. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 2595–2609. [[CrossRef](#)] [[PubMed](#)]
59. Zareapoor, M.; Shamsolmoali, P. Application of credit card fraud detection: Based on bagging ensemble classifier. *Procedia Comput. Sci.* **2015**, *48*, 679–685. [[CrossRef](#)]
60. Dong, L.; Yuan, Y.; Cai, Y. Using Bagging classifier to predict protein domain structural class. *J. Biomol. Struct. Dyn.* **2006**, *24*, 239–242.
61. Alkhamash, E.H.; Assiri, S.A.; Nemenqani, D.M.; Althaqafi, R.M.; Hadjouni, M.; Saeed, F.; Elshewey, A.M. Application of Machine Learning to Predict COVID-19 Spread via an Optimized BPSO Model. *Biomimetics* **2023**, *8*, 457. [[CrossRef](#)]
62. Fouad, Y.; Osman, A.M.; Hassan, S.A.; El-Bakry, H.M.; Elshewey, A.M. Adaptive Visual Sentiment Prediction Model Based on Event Concepts and Object Detection Techniques in Social Media. *Int. J. Adv. Comput. Sci. Appl.* **2023**, *14*, 252–256. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.