*mathematics*

MDPI

*Article*

# Machine Learning-Based Prediction Models of Acute Respiratory Failure in Patients with Acute Pesticide Poisoning

**Yeongmin Kim [1,†], Minsu Chae [2,†], Namjun Cho [3], Hyowook Gil [3] and Hwamin Lee [2,*]**

[1] Department of Computer Software Engineering, Soonchunhyang University, Asan 31538, Republic of Korea
[2] Department of Medical Informatics, College of Medicine, Korea University, Seoul 02841, Republic of Korea
[3] Department of Internal Medicine, Soonchunhyang University Cheonan Hospital, Cheonan 31151, Republic of Korea
* Correspondence: hwamin@korea.ac.kr; Tel.: +82-2-3407-2099
† These authors contributed equally to this work.

**Abstract:** The prognosis of patients with acute pesticide poisoning depends on their acute respiratory condition. Here, we propose machine learning models to predict acute respiratory failure in patients with acute pesticide poisoning using a decision tree, logistic regression, and random forests, support vector machine, adaptive boosting, gradient boosting, multi-layer boosting, recurrent neural network, long short-term memory, and gated recurrent gate. We collected medical records of patients with acute pesticide poisoning at the Soonchunhyang University Cheonan Hospital from 1 January 2016 to 31 December 2020. We applied the k-Nearest Neighbor Imputer algorithm, MissForest Impuer and average imputation method to handle the problems of missing values and outliers in electronic medical records. In addition, we used the min–max scaling method for feature scaling. Using the most recent medical research, *p*-values, tree-based feature selection, and recursive feature reduction, we selected 17 out of 81 features. We applied a sliding window of 3 h to every patient's medical record within 24 h. As the prevalence of acute respiratory failure in our dataset was 8%, we employed oversampling. We assessed the performance of our models in predicting acute respiratory failure. The proposed long short-term memory demonstrated a positive predictive value of 98.42%, a sensitivity of 97.91%, and an F1 score of 0.9816.

**Keywords:** machine learning; respiratory failure; acute pesticide poisoning; logistic regression; random forests; long short-term memory

**MSC:** 68T07; 9A16; 4008



**Citation:** Kim, Y.; Chae, M.; Cho, N.; Gil, H.; Lee, H. Machine Learning-Based Prediction Models of Acute Respiratory Failure in Patients with Acute Pesticide Poisoning. *Mathematics* **2022**, *10*, 4633. https://doi.org/10.3390/math10244633

Academic Editors: Ximeng Liu and Jose Luis Vicente Villardon

Received: 23 September 2022
Accepted: 4 December 2022
Published: 7 December 2022



## 1. Introduction

Pesticide toxicosis is caused by the ingestion of or exposure to pesticides [1]. In the Republic of Korea, the death toll from toxicosis is 2702 people and 1675 of the 2702 people (61.99%) had toxicosis caused by the ingestion of pesticides [2]. In addition, 71% of patients with pesticide poisoning have been reported to die within 6–24 h [2]. The most common reason for the ingestion of pesticides is suicide [3]. Each year, 110,000 individuals die from pesticide poisoning [3], which accounts for 13.7% of all suicides [3]. In the Republic of Korea, some regions (Chungcheong-do, Gangwon-do, Jeolla-do) have a higher death rate from pesticide poisoning than that the capital area (Seoul, Incheon, and Gyeonggi-do) [2]. Pesticide toxicosis is easily accessible, especially in these regions [2]. Neurological, respiratory, and cardiovascular symptoms have been reported in cases of pesticide toxicosis [1]. The prognosis of pesticide toxicosis depends on the extent of respiratory failure [1]. Respiratory failure is associated with a high death rate in hospitals. Current respiratory failure treatment options can be ineffective [4]. Preventing the failure of multiple organs is crucial in reducing the rate of mortality from respiratory failure [5]. Therefore, the prediction of respiratory failure is important for patient prognosis.

Recent predictions of respiratory failure include predicting respiratory failure based on semi-supervised learning [4]; predicting respiratory failure with clinical data [5]; predicting respiratory failure in patients with coronavirus disease-2019 (COVID-19) [6]; predicting respiratory failure in the intensive care unit (ICU) [7,8]; predicting respiratory failure in pesticide intoxication [9]; and predicting respiratory failure with simple patient trajectories [10].

Machine learning algorithms such as semi-recurrent neural networks (RNNs), extreme gradient boosting, logistic regression (LR), random forest (RF), and long short-term memory (LSTM) have been used to predict respiratory failure in previous studies. In the case of respiratory failure prediction based on semi-RNNs, the positive predictive value (PPV) was 3.3% and the sensitivity was 78.0% [4]. In the case of respiratory failure prediction with clinical data, the sensitivity was 71% [5]. In the case of respiratory failure prediction in patients with COVID-19, the PPV was 74% and the sensitivity was 78% [6]. In the case of respiratory failure prediction in the ICU, the PPV was 42% and the sensitivity was 80% [7]. In the case of respiratory failure prediction in pesticide intoxication, the PPV was 83.3% and the sensitivity was 60.6% [9]. In the case of respiratory failure prediction with simple patient trajectories, the PPV was 22.6% and the sensitivity was 88.1% [10]. Therefore, the performance of algorithms for predicting acute respiratory failure is low.

Our goal is to predict the prognosis for patients with acute pesticide poisoning. However, it is difficult to predict the prognosis because of the various causes of acute pesticide poisoning. We predict acute respiratory failure, an important prognostic factor for patients with acute pesticide poisoning. We predict acute respiratory failure within 24 h using machine learning and three-hour electronic medical records (EMRs) for patients. We perform EMR preprocessing as follows: (1) solve human errors; (2) solve missing values; (3) sliding window; (4) feature selection; (5) data scaling; and (6) solve the imbalance. Data preprocessing is important to improve performance [11–14]. Using current patient data to fill in the gaps, we imputed missing values using the k-Nearest Neighbor (KNN) imputer algorithm from scikit-learn [15], the MissForest Imputer, or the data average imputation technique. We used a sliding window dataset based on 3 h data. We performed feature selection based on the current medical knowledge and $p$-values and oversampling. We performed data scaling using MinMaxScaler provided by scikit-learn [15]. For predicting acute respiratory failure in acute pesticide poisoning, we utilize shallow learning such as decision tree (DT), random forest (RF), logistic regression (LR), support vector machine (SVM), adaptive boosting (AB), gradient boosting (GB), and deep learning such as multilayer perceptron (MLP), recurrent neural network (RNN), long short-term memory (LSTM), and gated recurrent unit (GRU).

## 2. Materials

### *2.1. Data*

This retrospective cohort study consisted of patients admitted to Soonchunhyang University Cheonan Hospital in the Republic of Korea between January 2016 and December 2020. The patients were over 19 years of age. The patients with pesticide poisoning and respiratory failure within 1 h of admission were excluded. The number of patients was 707. The pesticide categories included glyphosate, glufosinate, paraquat, organophosphate, pyrethroid, and carbamate. After replacing missing data, we performed sliding window data preprocessing, feature selection, and oversampling on the medical records.

When the data preprocessing process was completed, the total data consisted of 11,526 data with 17 features for 3 h. We split the data into the training dataset and test dataset at a 7:3 ratio. The training dataset was then divided with a 7:3 ratio into a training dataset and a test dataset. The training dataset was then divided with a 7:3 ratio into a training dataset and a holdout fold. The training dataset was then divided with a 7:3 ratio into a training dataset and a validation dataset. Using the training dataset, machine learning methods were constructed and evaluated using the holdout fold. The number of respiratory failures was only 909. Oversampling or undersampling can be used to solve the

imbalance in the datasets. We used oversampling algorithms such as the synthetic minority oversampling technique (SMOTE), borderline-SMOTE, and adaptive synthetic (ADASYN) given the limited cases of respiratory failure. Figure 1 shows the processing of patient selection. The number of training datasets was 7291, the number of validation set was 1695, and the number of holdout fold was 2421, and the number of test datasets was 3458.

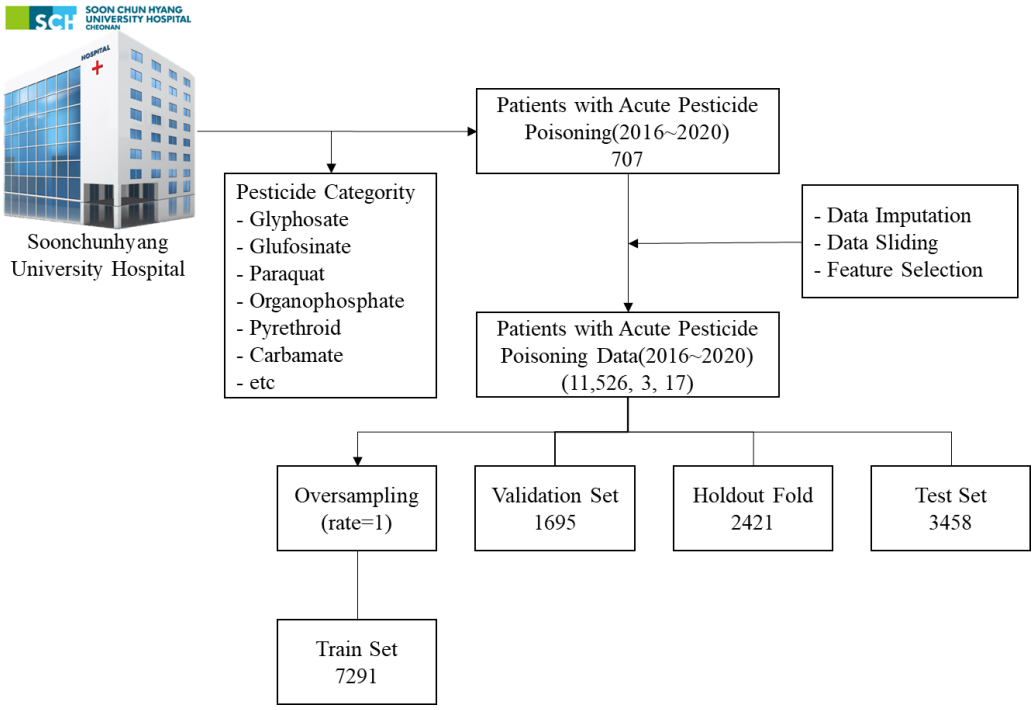


**Figure 1.** Patient selection, data preprocessing, and dataset. The training dataset number was 7291, and the validation set number was 1695, and the holdout fold number was 2421, and the test dataset number was 3458.

### *2.2. Replacement Missing Value*

Medical records are not free from missing values. The patient may be absent or there may be problems with noise or human errors. To improve machine learning algorithms, missing values need to be solved [11,12]. We solved missing values with the following three steps: (1) replace missing values with the recent data imputer (RDI); (2) apply the KNN imputation algorithm of scikit-learn with highly relevant features; and (3) replace other features through average imputation.

#### 2.2.1. Recent Data Imputer

Time-series data have continuous values over time. The RDI can be used to replace the missing values of each patient with recent data. Figure 2 shows the RDI algorithm. The average imputer, the maximum imputer, and the minimum imputer are non-consistent time-series characteristics. The RDI has time-series characteristics.

#### 2.2.2. k-Nearest Neighbor (KNN) Imputer

There may be missing values after using the RDI. The KNN imputer replaces missing values through distance functions for highly correlated features [13]. Improved performance may be achieved with the KNN compared with that of other imputers, such as the average imputer, maximum imputer, and minimum imputer [13]. We performed KNN imputation with highly relevant features twice as follows: (1) total $CO_2$, pH, $HCO_3$ standard, base excess, and lactate features; (2) $pCO_2$ and $pO_2$ features.

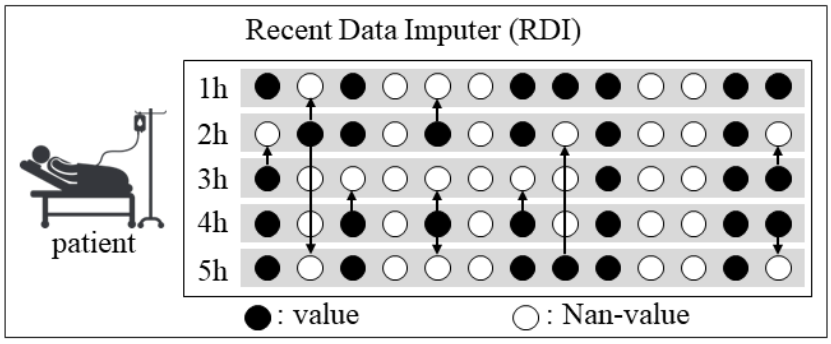


**Figure 2.** Processing by the recent data imputer (RDI). The black nodes indicate measured values and the white nodes indicate missing values.

#### 2.2.3. MissForest Imputer

The MissForest is an imputer based on RF [16]. Figure 3 shows the processing of MissForest. The MissForest replaces missing values to the median and trains on the dataset to interpolate missing values into prediction results.

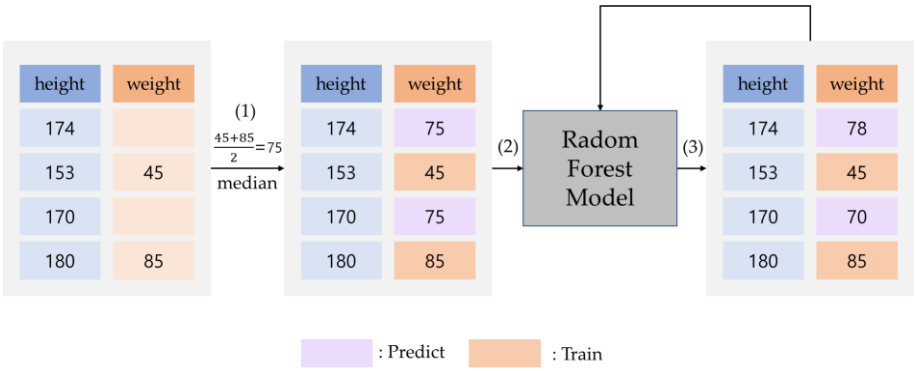


**Figure 3.** Processing by the MissForest. (1) Interpolate missing values into the median. (2) Train via dataset. (3) Interpolate missing values into the prediction results.

We interpolate total $CO_2$, pH, $HCO_3$ standard, base excess lactate, $pCO_2$, and $pO_2$ after performing the RDI algorithm.

### *2.3. Feature Selection*

Feature selection is important to improve the machine learning algorithm [14]. We perform according to a feature selection. (1) We calculate *p*-values to confirm unrelated features. (2) We determine features based on current medical knowledge. (3) We analyze the importance of features using RF and GB. (4) Using recursive feature elimination, we analyze both high- and low-ranking features. (5) In low-ranking features, we compare performance results exclusive to each feature.

First, we calculate the *p*-value for each feature and respiratory failure using ordinary least squares. To calculate the *p*-value, we utilize the OLS method provided by statsmodels [17]. We ignored features with *p*-value above 0.05 because they are uncorrelated factors. Table 1 shows the *p*-values for each feature.

We confirm that the features of *p*-values above 0.05 are as follows: smoking; alcohol; cardiovascular disease; SBP max; DBP max; RR max; Hb; glucose; BUN; creatinine; $pCO_2$; $HCO_3$ standard; BE; and troponin.

**Table 1.** The respiratory failure was correlated to *p*-values of features.

| Features | *p*-Value | Features | *p*-Value |
|---|---|---|---|
| Pesticide dose | 0.000 | WBC | 0.000 |
| Sex | 0.000 | PLT | 0.000 |
| Age | 0.000 | Albumin | 0.000 |
| BMI | 0.023 | Glucose | 0.070 |
| Smoking | 0.326 | BUN | 0.622 |
| Alcohol | 0.313 | Creatinine | 0.100 |
| Diabetes disease | 0.000 | Total $CO_2$ | 0.000 |
| Respiratory disease | 0.000 | C-reactive protein 1 | 0.000 |
| Cardiovascular disease | 0.995 | pH | 0.000 |
| GCS | 0.000 | $pCO_2$ | 0.199 |
| SBP max | 0.088 | $pO_2$ | 0.000 |
| DBP max | 0.897 | $O_2$ saturation | 0.000 |
| HR max | 0.000 | $HCO_3$ standard | 0.356 |
| RR max | 0.174 | BE | 0.120 |
| BT max | 0.000 | Troponin | 0.555 |
| Hb | 0.221 | Lactate | 0.000 |

BMI: body mass index; GCS: Glasgow Coma Scale; SBP: systolic blood pressure; DBP: diastolic blood pressure; HR: heart rate; RR: respiratory rate; BT: body temperature; max: maximum; Hb: hemoglobin; WBC: white blood cell; PLT: platelet; BUN: blood urea nitrogen; BE: base excess.

Second, we perform feature selection based on current medical knowledge. The following features are determined based on current medical knowledge: pesticide category; pesticide dose; sex; age; GCS; SBP max; HR max; BT max; WBC; PLT; albumin; total $CO_2$; CRP1; pH; $pO_2$; $O_2$ saturation; and lactate.

Third, we analyze tree-based feature selection methods. Table 2 shows the performance results of RF and GB. We confirm that sex is an unimportant feature.

**Table 2.** Performance result of tree-based feature selection method.

| Features | Tree-Based Feature Selection | | Features | Tree-Based Feature Selection | |
|---|---|---|---|---|---|
| | RF | GB | | RF | GB |
| Pesticide category | 0.056 | 0.073 | PLT | 0.032 | 0.013 |
| Pesticide dose | 0.032 | 0.017 | Albumin | 0.034 | 0.008 |
| Sex | 0.001 | 0.0003 | total_$CO_2$ | 0.072 | 0.025 |
| Age | 0.039 | 0.019 | C-reactive_protein_1 | 0.053 | 0.057 |
| GCS | 0.053 | 0.068 | pH | 0.175 | 0.247 |
| SBP_max | 0.010 | 0.003 | $pO_2$ | 0.018 | 0.003 |
| HR_max | 0.087 | 0.097 | $O_2$_saturation | 0.062 | 0.017 |
| BT_max | 0.033 | 0.029 | Lactate | 0.017 | 0.006 |
| WBC | 0.224 | 0.316 | | | |

RF: Random Forest; GB: Gradient Boost; GCS: Glasgow Coma Scale; SBP: systolic blood pressure; max: maximum; HR: heart rate; BT: body temperature; WBC: white blood cell; PLT: platelet.

Fourth, using recursive feature elimination, we analyze high- and low-ranking features. We perform recursive feature elimination based on SVM with linear, LR, RF, DT, and GB. We perform recursive feature elimination provided by scikit-learn. Table 3 shows the performance results of recursive feature elimination based on SVM with linear, LR, DT, and GB. We confirm that sex, albumin, and PLT are unimportant factors.

Fifth, we compare the performance results of each feature based on Tables 2 and 3. Table 4 shows the performance results of each feature using RF, GB, and MLP. The feature of the second stage is the highest performance in Table 4. We separate our dataset into train and test datasets. In the case of RF and GB, we train algorithms using stratified k-folds by train folds after separating train datasets into train folds and holdout folds. We train MLP algorithms using train folds and early stop using validation folds after separating

train datasets into the train, validation, and holdout folds. In addition, we compare the performance of each feature via each algorithm using holdout folds.

**Table 3.** Performance result of recursive feature elimination of each algorithm. Low-rank features are albumin, platelet, and sex. High-rank feature is pH.

| Machine Learning Algorithm | Low-Rank Feature | High-Rank Feature |
|---|---|---|
| SVM with linear | Albumin | pH |
| LR | PLT | pH |
| RF | Sex | pH |
| DT | Albumin | pH |
| GB | Sex | pH |

SVM: support vector machine; LR: logistic regression; RF: random forest; DT: decision tree; GB: gradient boost; PLT: platelet.

**Table 4.** Performance result of feature selection based on RF, GB, and MLP.

| Feature | Algorithm | PPV | Sensitivity | F1 Score | AUC |
|---|---|---|---|---|---|
| Reference | RF | 98.92% | 96.34% | 0.9761 | 0.9812 |
| | GB | 96.81% | 95.29% | 0.9604 | 0.9751 |
| | MLP | 96.81% | 95.29% | 0.9604 | 0.9751 |
| Reference exclude sex | RF | 98.92% | 95.81% | 0.9734 | 0.9786 |
| | GB | 92.78% | 94.24% | 0.9351 | 0.9681 |
| | MLP | 92.78% | 94.24% | 0.9351 | 0.9681 |
| Reference exclude PLT | RF | 99.46% | 95.81% | 0.9760 | 0.9788 |
| | GB | 98.89% | 93.19% | 0.9596 | 0.9655 |
| | MLP | 92.78% | 94.24% | 0.9351 | 0.9681 |
| Reference exclude albumin | RF | 98.39% | 95.81% | 0.9708 | 0.9784 |
| | GB | 96.70% | 92.15% | 0.9437 | 0.9594 |
| | MLP | 96.70% | 92.15% | 0.9437 | 0.9594 |

PPV: positive predictive value; RF: random forest; GB: gradient boosting; MLP: multi-layer perceptron; PLT: platelet.

For the performance evaluation of each feature, we perform steps one through five. The first and second steps exhibit the highest performance when each feature's performance is evaluated. Therefore, we used the following features: pesticide category; pesticide dose; sex; age; GCS; SBP max; HR max; BT max; WBC; PLT; albumin; total $CO_2$; CRP1; pH; $pO_2$; $O_2$ saturation; and lactate.

### *2.4. Hour Sliding Window in 24 H*

In this study, our objective was to predict respiratory failure within 24 h using 3 h data. Figure 4 shows the sliding window to time-series data in 24 h based on 3 h data.

### *2.5. MinMaxScaler*

In machine learning, each feature unit is different; thus, the results can be biased. To solve this, it is necessary to use the same data range. In this study, we used the MinMaxScaler provided by scikit-learn [15]. It expresses a value between 0 and 1 through Equation (1).

$$X = \frac{X - \min}{\max - \min} \quad (1)$$

The X variable represents a feature in the dataset. The min variable represents the minimum value of each feature. The max variable represents the maximum value of each feature.

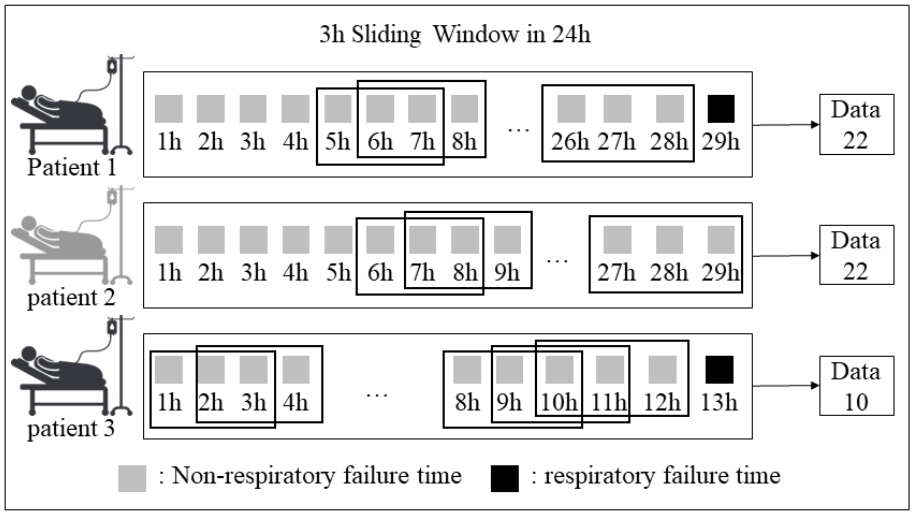


**Figure 4.** Processing by sliding window.

### *2.6. Oversampling*

In this study, the prevalence of respiratory failure was 8%, which suggests an imbalance. There are two methods of solving the imbalance: (1) oversampling and (2) undersampling. In this study, the number of cases of respiratory failure was limited and oversampling rather than undersampling tends to be better for improving performance [18]. Therefore, we performed oversampling. We applied SMOTE, borderline-SMOTE, and ADADYN to solve the data imbalance [19].

#### 2.6.1. Synthetic Minority Oversampling Technique (SMOTE)

The SMOTE algorithm was proposed by Chawla et al. [20]. The SMOTE algorithm has three steps as follows: (1) apply the KNN algorithm in the minority class after randomized minority data selection [20]; (2) choose randomized minority data in the nearest data [20]; and (3) locate generated data between one-step and two-step data. Figure 5 shows the processing of the SMOTE algorithm.

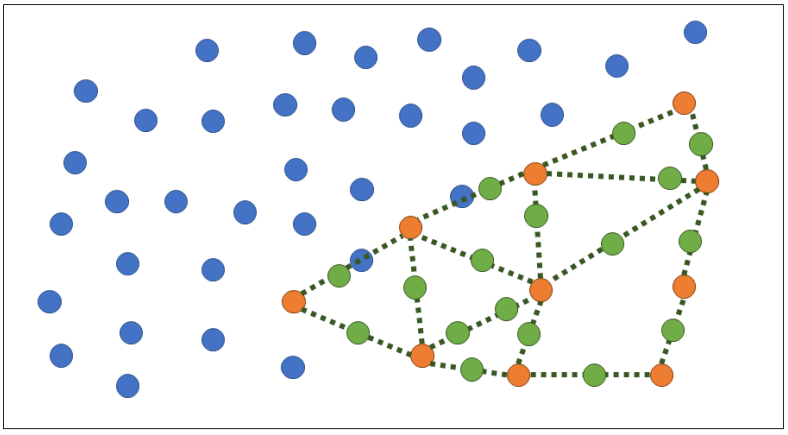

**Figure 5.** SMOTE processing. Blue indicates the majority dataset (non-respiratory failure). Orange indicates the minority dataset (respiratory failure). Green indicates the minority dataset (respiratory failure), which is generated by SMOTE.

#### 2.6.2. Borderline-SMOTE

The borderline-SMOTE algorithm is based on the original SMOTE [21]. Notably, minority data may be far from the minority data-majority data boundary [21]. The borderline-SMOTE is applied at the boundary between minority data and majority data. After the

KNN algorithm is applied to the minority data, the borderline-SMOTE algorithm considers data comprising more than half of the data to be borderline. The borderline-SMOTE applies the SMOTE algorithm to minority data at the borderline. Figure 6 shows data processing by the borderline-SMOTE, which generates minority data at the borderline and achieves classifier efficiency [19,21].

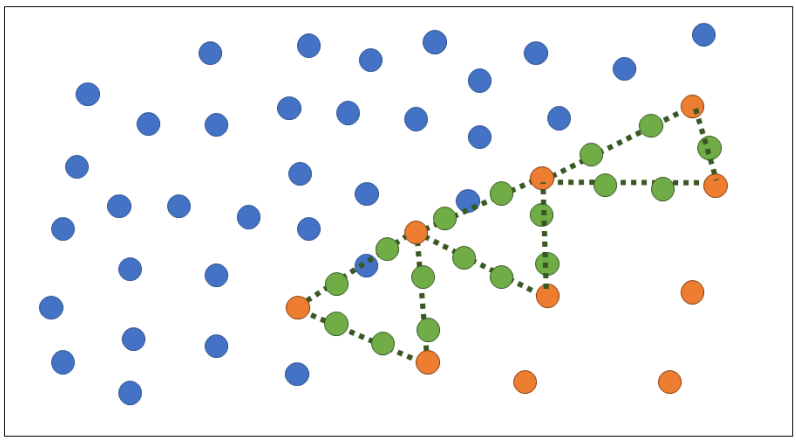

**Figure 6.** Borderline-SMOTE processing. Blue indicates the majority dataset (non-respiratory failure). Orange indicates the minority dataset (respiratory failure). Green indicates the minority dataset (respiratory failure) generated by Borderline-SMOTE.

#### 2.6.3. Adaptive Synthetic (ADASYN)

The ADASYN algorithm applies the KNN algorithm to minority data to generate minority data if there is a huge amount of majority data [19]. The ADASYN algorithm consists of four steps as follows [19]: (1) KNN algorithm calculates the ratio of minority data to majority data for each minority datum [19]; (2) the sum of the ratios of majority data is divided by each ratio of majority data [19]; (3) it is calculated to repeat through Equation (2) [19]; and (4) generate minority data as much as a repeat on each minority dataset [19]. Figure 7 shows data processing by the ADASYN algorithm.

$$\text{repeat} = \text{second step result} \times (\text{number of majority} - \text{number of minority}) \quad (2)$$

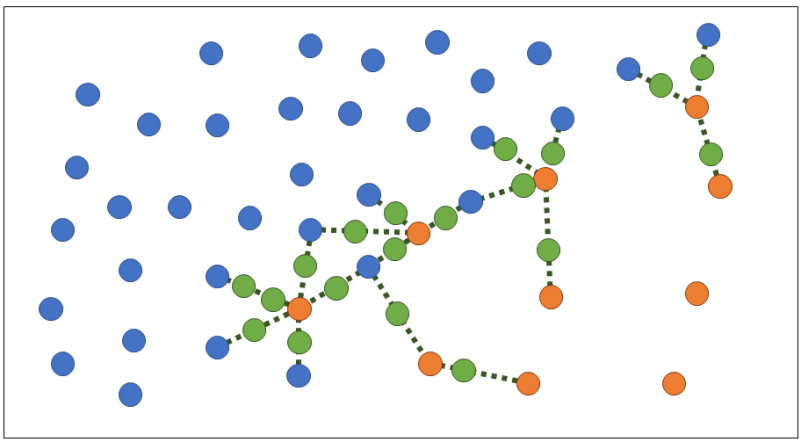

**Figure 7.** ADASYN processing. Blue indicates the majority dataset (non-respiratory failure). Orange indicates the minority dataset (respiratory failure). Green indicates the minority dataset (respiratory failure) generated by ADASYN. When there are more majority data surrounding minority data, more minority data are produced.

## 3. Methods

### *3.1. Shallow Learning*

A time-series dataset has three dimensions: number of datasets, amount of time, and number of features, e.g., (11,148, 3, 17). However, the machine learning algorithms provided by scikit-learn [15] use only two dimensions. We used both time-series and non-time-series features. The time-series features were the maximum systolic blood pressure (SBP) in 1 h, maximum heart rate (HR) in 1 h, maximum body temperature (BT) in 1 h, white blood cell (WBC), platelet (PLT), albumin, total $CO_2$, cysteine-rich protein 1 (CRP1), pH, $pO_2$, $O_2$ saturation, and lactate. The non-time-series features were pesticide category, pesticide dose, sex, age, and Glasgow Coma Scale (GCS). We expressed the dataset structure as the number of datasets and number of features, e.g., (11,148, 41).

#### 3.1.1. Logistic Regression (LR)

LR classification is based on the sigmoid function. LR calculates the weight and bias in the training dataset [22]. In this study, LR was used to calculate z with 41 features through Equation (3) [22].

$$Z = \sum_{i=1}^{41} w_i a_i + b \tag{3}$$

LR was used to calculate the sigmoid function with z through Equation (4) [22].

$$F(z) = \frac{1}{1 + e^{-z}} \tag{4}$$

LR classification was performed by f(z) through Equation (5) [22]. If f(z) is greater than 0.5, it is classified as respiratory failure; otherwise, it indicates non-respiratory failure. We utilized the LR algorithm provided by scikit-learn [15].

$$\text{predict} = \begin{cases} 0 \text{ if } f(z) \leq 0.5 \\ 1 \text{ if } f(z) > 0.5 \end{cases} \tag{5}$$

#### 3.1.2. Decision Tree (DT)

DT utilizes a binary tree structure, which can classify highly related features of respiratory failure [3]. DT calculates impurity and branches until the leaf node impurity is 0. There are two methods of calculating impurity: (1) the Gini coefficient and (2) the entropy coefficient. In this study, we calculated impurity using the Gini coefficient through Equation (6) [15].

$$\text{Gini} = \sum_{i=1}^{n} (R_i(1 - \sum_{k=1}^{m} {P_k}^2)) \tag{6}$$

The n variable represents the number of nodes and the m variable represents the number of outcomes. In this study, m is 2. The $R_i$ variable represents the sample ratio of each branch. The $P_k$ variable represents the class ratio. DT utilizes pruning to solve overfit. Figure 8 shows the DT algorithms. The 1 h prefix indicates the medical record after 1 h of measurement. The 2 h prefix indicates the medical record after 2 h of measurement. The 3 h prefix indicates the medical record after 3 h of measurement. We used the DT model provided by scikit-learn [15].

#### 3.1.3. Random Forest (RF)

RF is an ensemble model that uses many DTs and bootstrap aggregation [23]. Figure 9 shows the processing of RF. After each DT in the RF predicts respiratory failure, the RF classifier votes on the prediction. In this study, we used a max depth hyperparameter of 6 and a n_estimators hyperparameter of 100 in the RandomForestClassifier provided by scikit-learn [15].

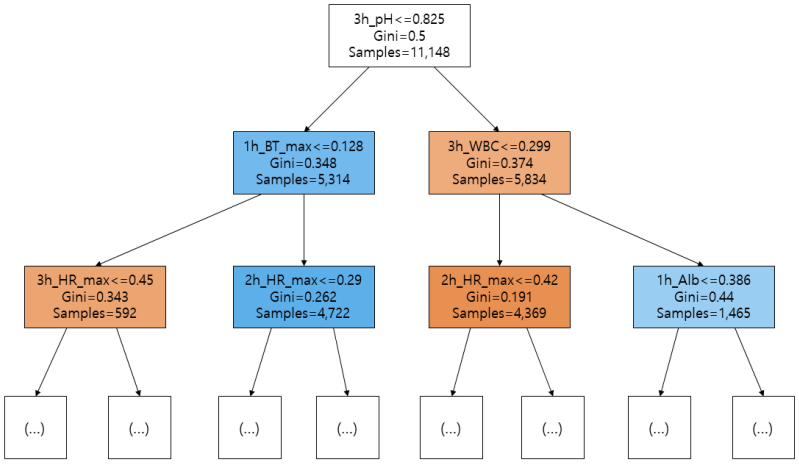


**Figure 8.** Decision tree. Orange indicates the prediction of respiratory failure in each node. Blue indicates non-respiratory failure. Darker nodes contain more data.

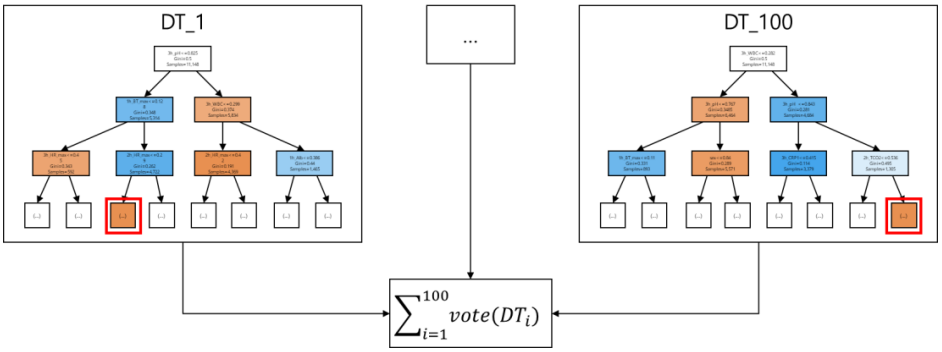


**Figure 9.** Processing of random forests. The red box in the decision tree indicates the predicted outcome. The random forest classifier is based on decision tree outcomes. Random Forest performs bootstrap aggregation on many decision trees and uses voting to determine the prediction results of many decision trees.

### 3.1.4. Support Vector Machine (SVM)

SVM is a support vector base classifier and not a decision boundary [24]. There is a problem with overfitting when other labels are near to the decision boundary. A support vector is a calculated boundary that minimizes classification error. Figure 10 shows classification based on the support vector of SVM.

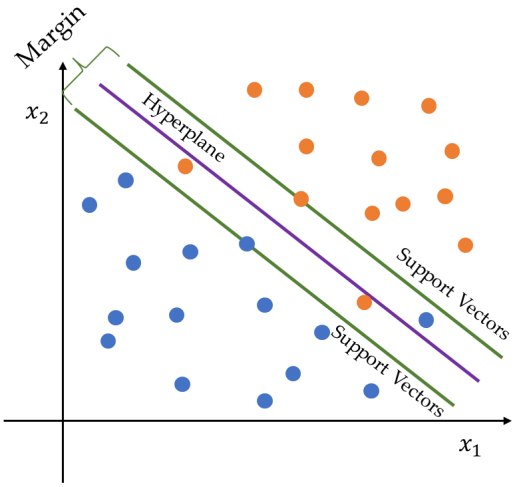


**Figure 10.** It shows the classification of SVM.

#### 3.1.5. Adaptive Boost (AB)

AB has two leaf nodes named as stump [25]. AB reduces the error by providing the next estimator with the weight of the incorrectly predicted respiratory dataset through the stump [25]. Figure 11 shows the AB weight calculation process.

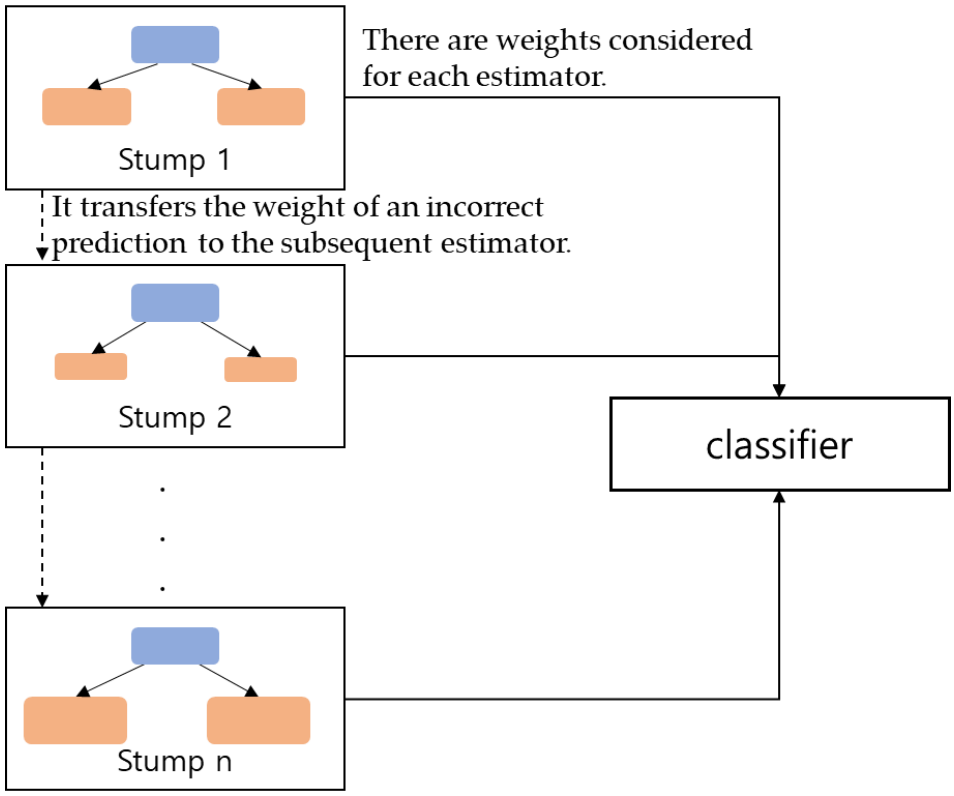


**Figure 11.** Processing of adaptive boost. Each estimator transfers the weight of an incorrect prediction after training via the dataset.

#### 3.1.6. Gradient Boost (GB)

The GB calculates the residual error for solving an incorrect prediction result by subtracting the measured value from the prediction value. GB calculates the residual error based on the tree using actual respiratory failure and prediction results.

### *3.2. Deep Learning Algorithms*

We organize datasets in three dimensions because RNN-based models support time-series data. We utilize TensorFlow to perform deep learning [26].

#### 3.2.1. Multi-Layer Perceptron (MLP)

The perceptron calculates the weight of many features such as EMRs, for prediction. However, single perceptrons suffer with non-linear datasets. In order to solve the nonlinear problem, the perceptron organizes multiple layers, which is referred to as MLP. The MLP does not support three dimensions, so we organize datasets in two dimensions. We implement MLP utilizing dense layers provided by TensorFlow [26]. Figure 12 shows the structure of MLP.

#### 3.2.2. Recurrent Neural Network (RNN)

Without considering the order of the time series, machine learning recognizes the features of time-series as other features [27]. An RNN model has been proposed for considering the order of the time series. The vanilla RNN model calculates the current weights using

the weights from the previous EMRs and the current EMRs. We implement RNN utilizing a simple RNN layer provided by TensorFlow. Figure 13 shows the structure of RNN.

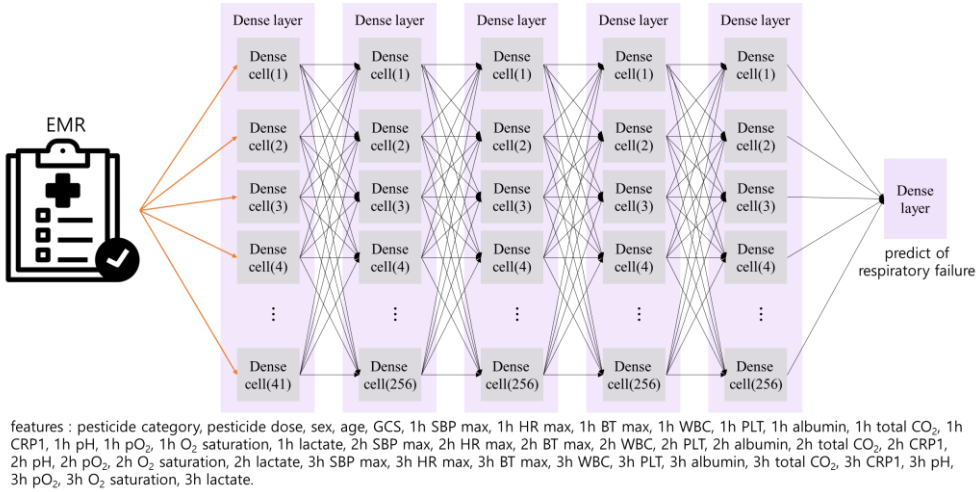


**Figure 12.** Structure of MLP in this paper. The number of units in hidden layer is 256.

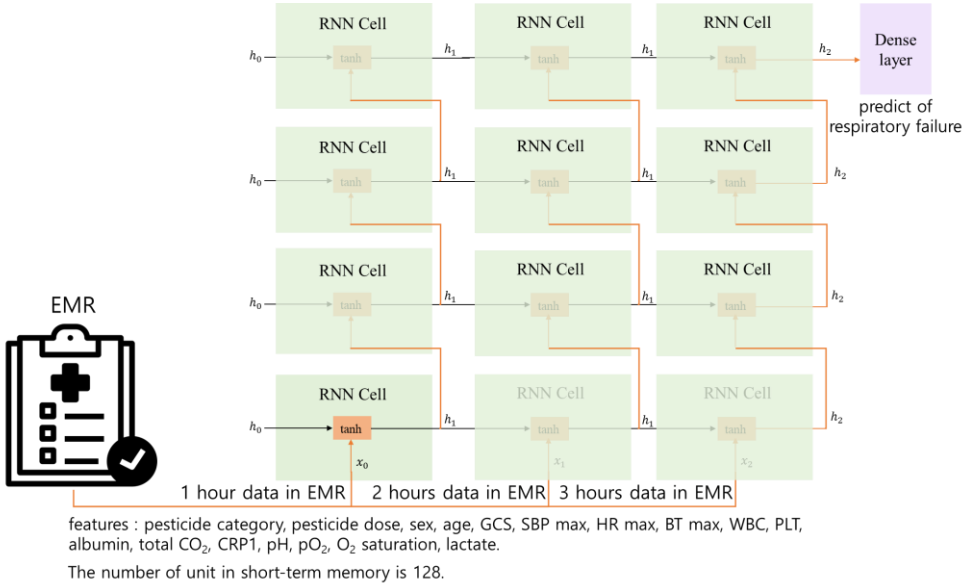


**Figure 13.** Structure of the RNN model in this paper. The number of units in the RNN layer is 128.

### 3.2.3. Long Short-Term Memory (LSTM)

The vanilla RNN has the problem of considering only previous and current EMRs [28]. LSTM organizes forget gate, input gate, and output gate to solve short-term dependent problems [28]. The input gate calculates reminder information for long-term memory using the weight of previous short-term memory and current EMRs. The forget gate calculates the removal information for long-term memory using the weight of previous short-term memory and current EMRs. The long-term memory updates using the results of the forget gate and input gate. The output gate calculates weight using previous short-term memory and current EMRs and current long-term memory. We utilize the LSTM layer provided by TensorFlow. Figure 14 shows the structure of LSTM.

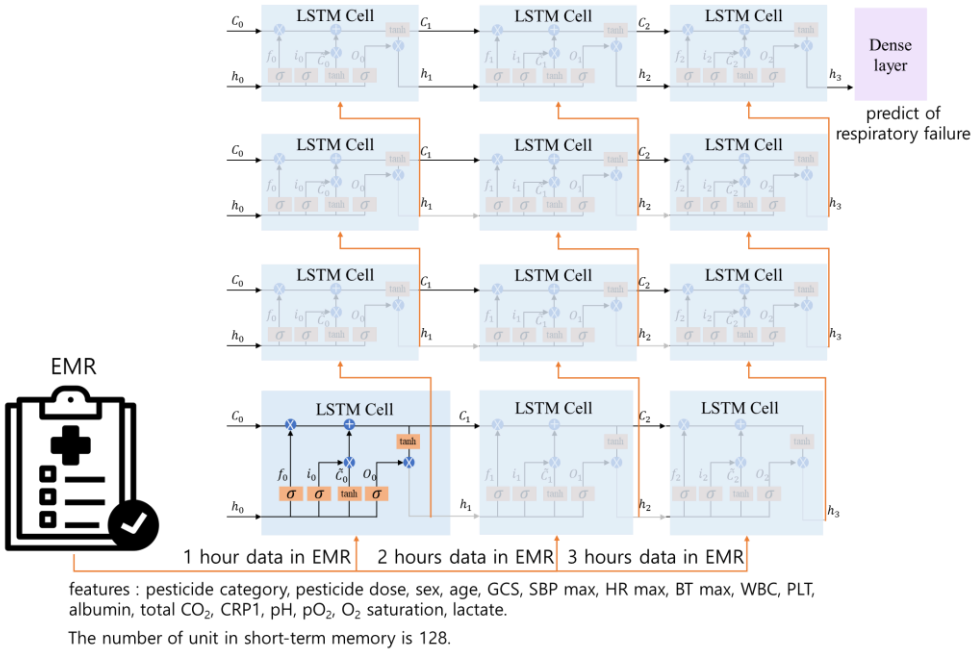


**Figure 14.** Structure of the LSTM model in this paper. The number of units in the LSTM layer is 128.

#### 3.2.4. Gated Recurrent Unit (GRU)

To solve long-term dependency in vanilla RNN, the GRU organizes update gates and reset gates [29]. The weight for the prediction of acute respiratory failure is calculated through previous memory and current EMRs. The update gates determine whether to use the previous memory or the current weight. The reset gates calculate the removal information for memory. We utilize the GRU layer provided by TensorFlow. Figure 15 shows the structure of GRU.

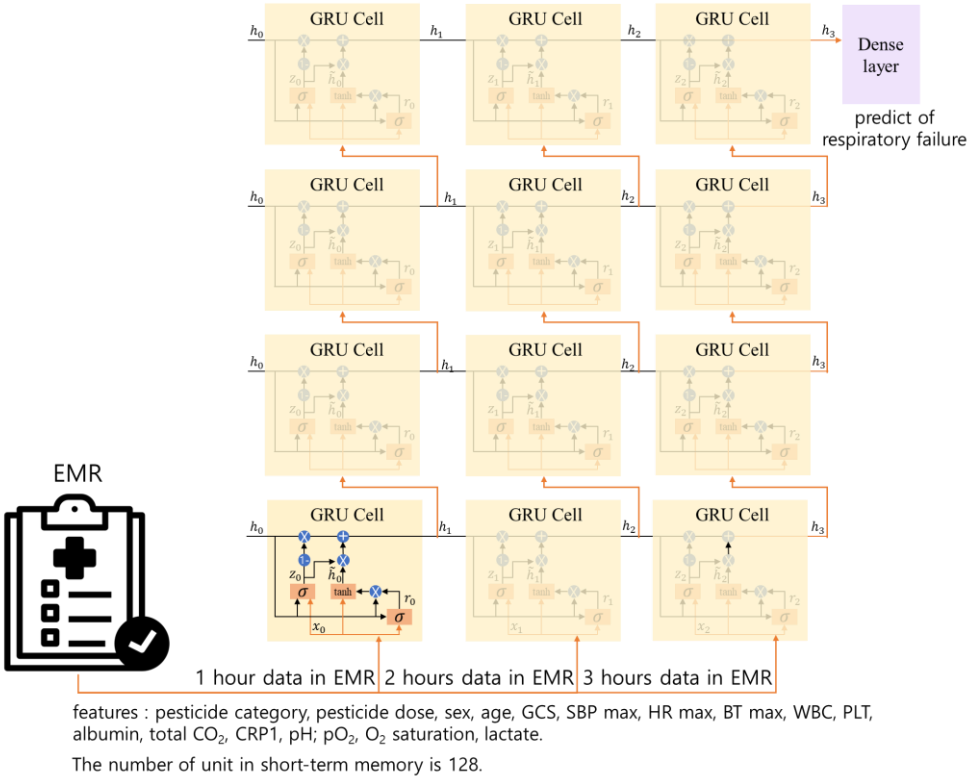


**Figure 15.** Structure of the GRU model in this paper. The number of units in the GRU layer is 128.

### *3.3. Stratified-k-Fold*

Cross-validation involves the splitting of the training dataset into a training fold and a test fold [30]. The training fold is used for learning in the training step [30]. The test fold is used for validation after the machine learning training step [30]. This technique can be used to solve overfitting [31]. The stratified k-fold is a cross-validation method that separates the data into k-training folds and test folds based on the class ratio (Figure 16).

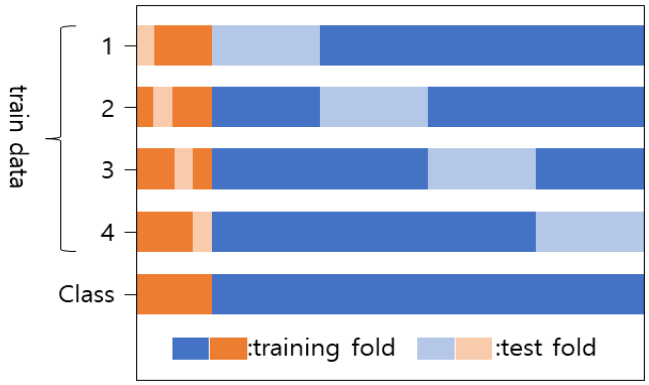


**Figure 16.** Stratified k-fold processing (in this case, k is 4).

## 4. Results

### *4.1. Evaluation Methods*

In machine learning, binary classification is mainly evaluated by a confusion matrix. Table 5 shows the confusion matrix. In this study, the patient dataset was imbalanced. Therefore, the PPV and sensitivity are more important than accuracy. The PPV indicates the rate of actual respiratory failure patients among patients with predicted respiratory failure, which can be calculated through Equation (7).

$$\text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{7}$$

**Table 5.** Confusion matrix.

| | | **Actual** | |
|---|---|---|---|
| | | Respiratory failure | Non-respiratory failure |
| **Precision** | Respiratory failure | True positive (TP) | False positive (FP) |
| | Non-respiratory failure | False negative (FN) | True negative (TN) |

The sensitivity indicates the rate of patients with predicted respiratory failure among actual patients with respiratory failure, which can be calculated through Equation (8).

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{8}$$

The F1 score considers both the PPV and sensitivity, which can be calculated through Equation (9).

$$\text{F1 score} = 2 \times \frac{\text{PPV} \times \text{Sensitivity}}{\text{PPV} + \text{Sensitivity}} \tag{9}$$

### *4.2. Characteristics of Study*

We used 17 features to predict respiratory failure. Table 6 shows the mean and standard deviation of each feature in this study. We excluded the "pesticide category" from the features because there were several pesticide categories. We construct time series

characteristics comprising 3 h measurements. To avoid overfitting, we oversampled the training dataset.

**Table 6.** The characteristics of our dataset.

| | **Training Data (*n* = 8068)** | **Test Data (*n* = 3458)** |
|---|---|---|
| Pesticide dose | 171.87 ± 138.07 | 177.18 ± 144.12 |
| Sex, male | 4994, 61.90% | 2210, 63.91% |
| Age | 61.07 ± 16.37 | 61.40 ± 16.48 |
| GCS | 13.90 ± 2.10 | 13.85 ± 2.19 |
| 1h_SBP_max | 124.00 ± 18.03 | 123.89 ± 18.16 |
| 1h_HR_max | 76.40 ± 14.37 | 76.30 ± 14.36 |
| 1h_BT_max | 36.55 ± 0.36 | 36.55 ± 0.36 |
| 1h_WBC | 7.53 ± 3.17 | 7.54 ± 3.15 |
| 1h_PLT | 144.52 ± 64.19 | 142.90 ± 65.09 |
| 1h_albumin | 3.59 ± 0.48 | 3.57 ± 0.49 |
| 1h_total_$CO_2$ | 24.14 ± 3.15 | 24.23 ± 3.15 |
| 1h_C-reactive_protein_1 | 26.97 ± 50.49 | 28.73 ± 51.61 |
| 1h_pH | 7.43 ± 0.06 | 7.43 ± 0.06 |
| 1h_$pO_2$ | 93.29 ± 26.62 | 93.02 ± 24.74 |
| 1h_$O_2$_saturation | 96.02 ± 3.67 | 96.03 ± 3.62 |
| 1h_lactate | 2.43 ± 2.02 | 2.48 ± 2.12 |
| 2h_SBP_max | 123.92 ± 17.90 | 123.93 ± 18.32 |
| 2h_HR_max | 76.40 ± 14.38 | 76.38 ± 14.43 |
| 2h_BT_max | 36.55 ± 0.35 | 36.56 ± 0.37 |
| 2h_WBC | 7.53 ± 3.17 | 7.54 ± 3.15 |
| 2h_PLT | 144.12 ± 63.80 | 142.47 ± 64.86 |
| 2h_albumin | 3.58 ± 0.48 | 3.57 ± 0.49 |
| 2h_total_$CO_2$ | 24.14 ± 3.15 | 24.23 ± 3.15 |
| 2h_C-reactive_protein_1 | 26.98 ± 50.49 | 28.74 ±51.61 |
| 2h_pH | 7.43 ± 0.06 | 7.43 ± 0.07 |
| 2h_$pO_2$ | 93.41 ± 26.20 | 92.98 ± 24.72 |
| 2h_$O_2$_saturation | 95.99 ± 3.79 | 96.00 ± 3.72 |
| 2h_lactate | 2.44 ± 2.03 | 2.48 ± 2.13 |
| 3h_SBP_max | 123.84 ± 18.01 | 123.79 ± 17.99 |
| 3h_HR_max | 76.48 ± 14.52 | 76.34 ± 14.36 |
| 3h_BT_max | 36.56 ± 0.35 | 36.56 ± 0.36 |
| 3h_WBC | 7.53 ± 3.17 | 7.53 ± 3.15 |
| 3h_PLT | 143.94 ± 63.64 | 142.42 ± 64.84 |
| 3h_albumin | 3.58 ± 0.48 | 3.57 ± 0.49 |
| 3h_total_$CO_2$ | 24.14 ± 3.16 | 24.24 ± 3.15 |
| 3h_C-reactive_protein_1 | 26.99 ± 50.48 | 28.74 ± 51.61 |
| 3h_pH | 7.43 ± 0.07 | 7.43 ± 0.07 |
| 3h_$pO_2$ | 93.44 ± 25.88 | 92.98 ± 24.74 |
| 3h_$O_2$_saturation | 95.99 ± 3.81 | 95.97 ± 3.88 |
| 3h_lactate | 2.44 ± 2.04 | 2.49 ± 2.15 |

GCS: Glasgow Coma Scale; SBP: systolic blood pressure; HR: heart rate; BT: body temperature; WBC: white blood cell; PLT: platelet.

### *4.3. Evaluation of Imputation*

We compared the performances of KNN and missForest after interpolating with RDI. We separate our dataset into train and test datasets. In the case of RF and GB, we train algorithms using stratified k-folds by train folds after separating train datasets into train folds and holdout folds. We train MLP algorithms using train folds and early stop using validation folds after separating train datasets into train, validation, and holdout folds. In addition, we compare the performance of the imputer via each algorithm using holdout folds. The results of KNN and missForest imputation using validation datasets based on RF, GB, and MLP are shown in Table 7.

**Table 7.** Performance comparison of KNN and missForest based on RF, GB, and MLP.

| Imputation | Algorithm | PPV | Sensitivity | F1 Score | AUC |
|---|---|---|---|---|---|
| KNN imputer | RF | 98.92% | 96.34% | 0.9761 | 0.9812 |
| | GB | 97.80% | 93.19% | 0.9544 | 0.9651 |
| | MLP | 96.81% | 95.29% | 0.9604 | 0.9751 |
| MissForest imputer | RF | 98.92% | 96.34% | 0.9761 | 0.9812 |
| | GB | 97.74% | 90.58% | 0.9402 | 0.9520 |
| | MLP | 96.17% | 92.15% | 0.9412 | 0.9592 |

PPV: positive predictive value; RF: random forest; GB: gradient boosting; MLP: multi-layer perceptron.

In this paper, we confirm KNN imputer outperforms the MissForest imputer. We perform the KNN imputer after interpolating through RDI.

### *4.4. Evaluation of Hyperparameter Tuning*

To improve the performance of each algorithm, we perform hyperparameter turning. We separate our dataset into train and test datasets. In the case of RF and GB, we train algorithms using stratified k-folds by train folds after separating train datasets into train folds and holdout folds. We train MLP algorithms using train folds and early stop using validation folds after separating train datasets into train, validation, and holdout folds. In addition, we compare the performance of hyperparameter tuning via each algorithm using holdout folds. On DT and RF, we tune the hyperparameter for the information gain method and max depth. We use SVM to turn hyperparameters for the penalty of square l2 and kernels with radial basis functions (RBF), linear, and polynomial. On AB, we tune the hyperparameters for the number of estimators and learning rate. On GB, we tune hyperparameters for the number of estimators and max depth. On MLP, RNN, LSTM, and GRU, we tune hyperparameters for the unit size and dropout rate. The performance results of each algorithm according to their hyperparameters are shown in Table 8.

**Table 8.** Performance comparison of hyperparameter turning.

| Algorithm | Hyperparameter | | | | PPV | Sensitivity | F1 Score | AUC |
|---|---|---|---|---|---|---|---|---|
| DT | Function of computational complexity | Gini | Max depth | 8 | 95.76% | 82.72% | 0.8876 | 0.9120 |
| | | | | 9 | 96.49% | 86.39% | 0.9116 | 0.9306 |
| | | | | 10 | 98.17% | 84.29% | 0.9070 | 0.9208 |
| | | Entropy | | 8 | 97.27% | 93.19% | 0.9519 | 0.9648 |
| | | | | 9 | 98.35% | 93.72% | 0.9598 | 0.9679 |
| | | | | 10 | 98.31% | 91.62% | 0.9485 | 0.9574 |
| RF | Function of computational complexity | Gini | Max depth | 8 | 98.66% | 76.96% | 0.8647 | 0.8844 |
| | | | | 9 | 98.76% | 83.25% | 0.9034 | 0.9158 |
| | | | | 10 | 98.74% | 82.20% | 0.8971 | 0.9105 |
| | | Entropy | | 8 | 98.86% | 91.10% | 0.9482 | 0.9550 |
| | | | | 9 | 98.92% | 96.34% | 0.9761 | 0.9812 |
| | | | | 10 | 98.91% | 95.29% | 0.9707 | 0.9760 |

**Table 8.** *Cont.*

| Algorithm | Hyperparameter | | | PPV | Sensitivity | F1 Score | AUC |
|---|---|---|---|---|---|---|---|
| SVM | Regularization parameter | 5.5 | kernel | RBF | 99.32% | 76.96% | 0.8673 | 0.8846 |
| | | | | Linear | 77.36% | 42.92% | 0.5522 | 0.7093 |
| | | | | Poly | 97.43% | 79.58% | 0.8761 | 0.8970 |
| | | 6.0 | | RBF | 99.33% | 77.49% | 0.8706 | 0.8872 |
| | | | | Linear | 76.58% | 44.50% | 0.5629 | 0.7167 |
| | | | | Poly | 96.84% | 80.10% | 0.8768 | 0.8994 |
| | | 6.5 | | RBF | 99.34% | 79.06% | 0.8805 | 0.8950 |
| | | | | Linear | 75.68% | 43.98% | 0.5563 | 0.7138 |
| | | | | Poly | 86.84% | 80.10% | 0.8768 | 0.8994 |
| AB | Number of estimators | 140 | Learning rate | 0.5 | 94.08% | 74.87% | 0.8338 | 0.8723 |
| | | | | 1.0 | 95.43% | 87.43% | 0.9126 | 0.9354 |
| | | | | 2.0 | 6.91% | 84.82% | 0.1277 | 0.4344 |
| | | 150 | | 0.5 | 94.34% | 78.53% | 0.8571 | 0.8907 |
| | | | | 1.0 | 96.07% | 89.53% | 0.9268 | 0.9461 |
| | | | | 2.0 | 6.91% | 84.82% | 0.1277 | 0.4344 |
| | | 160 | | 0.5 | 94.97% | 79.06% | 0.8629 | 0.8935 |
| | | | | 1.0 | 94.97% | 89.01% | 0.9189 | 0.9430 |
| | | | | 2.0 | 6.91% | 84.82% | 0.1277 | 0.4344 |
| GB | Number of estimators | 110 | Max depth | 3 | 95.73% | 82.99% | 0.8845 | 0.9094 |
| | | | | 4 | 97.77% | 91.62% | 0.9459 | 0.9572 |
| | | | | 5 | 97.80% | 93.19% | 0.9544 | 0.9651 |
| | | 120 | | 3 | 96.45% | 85.34% | 0.9056 | 0.9254 |
| | | | | 4 | 97.80% | 93.19% | 0.9544 | 0.9651 |
| | | | | 5 | 97.80% | 93.19% | 0.9544 | 0.9651 |
| | | 130 | | 3 | 96.45% | 85.34% | 0.9056 | 0.9254 |
| | | | | 4 | 97.78% | 92.15% | 0.9488 | 0.9598 |
| | | | | 5 | 96.55% | 87.96% | 0.9205 | 0.9384 |
| MLP | Unit size | 64 | Dropout rate | 20% | 97.13% | 88.48% | 0.9260 | 0.9413 |
| | | | | 30% | 96.63% | 90.05% | 0.9322 | 0.9489 |
| | | | | 40% | 95.71% | 81.68% | 0.8814 | 0.9068 |
| | | 128 | | 20% | 97.71% | 89.53% | 0.9344 | 0.9467 |
| | | | | 30% | 98.24% | 87.43% | 0.9252 | 0.9365 |
| | | | | 40% | 94.97% | 79.06% | 0.8629 | 0.8935 |
| | | 256 | | 20% | 96.81% | 95.29% | 0.9604 | 0.9751 |
| | | | | 30% | 97.80% | 93.19% | 0.9544 | 0.9651 |
| | | | | 40% | 92.73% | 80.10% | 0.8596 | 0.8798 |

**Table 8.** *Cont.*

| Algorithm | Hyperparameter | | | PPV | Sensitivity | F1 Score | AUC |
|---|---|---|---|---|---|---|---|
| RNN | Unit size | 64 | Dropout rate | 20% | 76.28 | 62.30% | 0.6859 | 0.8032 |
| | | | | 30% | 86.26% | 82.20% | 0.8418 | 0.9054 |
| | | | | 40% | 71.34% | 66.49% | 0.6883 | 0.8210 |
| | | 128 | | 20% | 98.32% | 92.15% | 0.9514 | 0.9601 |
| | | | | 30% | 97.85% | 95.29% | 0.9655 | 0.9755 |
| | | | | 40% | 78.08% | 59.69% | 0.6766 | 0.7913 |
| | | 256 | | 20% | 96.65% | 90.58% | 0.9351 | 0.9515 |
| | | | | 30% | 64.88% | 69.63% | 0.6717 | 0.8320 |
| | | | | 40% | 98.22% | 86.91% | 0.9222 | 0.9339 |
| LSTM | Unit size | 64 | Dropout rate | 20% | 95.72% | 93.72$ | 0.9471 | 0.9668 |
| | | | | 30% | 98.29% | 90.05% | 0.9399 | 0.9496 |
| | | | | 40% | 72.15% | 59.69% | 0.6533 | 0.7886 |
| | | 128 | | 20% | 96.15% | 91.62% | 0.9383 | 0.9565 |
| | | | | 30% | 98.29% | 90.05% | 0.9399 | 0.9459 |
| | | | | 40% | 98.88% | 92.67% | 0.9568 | 0.9629 |
| | | 256 | | 20% | 98.87% | 91.62% | 0.9511 | 0.9577 |
| | | | | 30% | 97.77% | 91.62% | 0.9459 | 0.9572 |
| | | | | 40% | 98.86% | 91.10% | 0.9482 | 0.9550 |
| GRU | Unit size | 64 | Dropout rate | 20% | 97.75% | 91.10% | 0.9431 | 0.9546 |
| | | | | 30% | 98.82% | 87.43% | 0.9278 | 0.9367 |
| | | | | 40% | 98.32% | 92.15% | 0.9514 | 0.9601 |
| | | 128 | | 20% | 98.88% | 92.15% | 0.9539 | 0.9603 |
| | | | | 30% | 97.16% | 89.53% | 0.9319 | 0.9465 |
| | | | | 40% | 97.30% | 94.24% | 0.9574 | 0.9701 |
| | | 256 | | 20% | 97.78% | 92.15% | 0.9488 | 0.9598 |
| | | | | 30% | 98.31% | 91.10% | 0.9457 | 0.9548 |
| | | | | 40% | 98.34% | 93.19% | 0.9570 | 0.9653 |

PPV: positive predictive value; DT: decision tree; RF: random forest; SVM: support vector machine; RBF: radial basis function; poly: polynomial; AB: adaptive boost; GB: gradient boost; MLP: multi-layer perceptron; RNN: recurrent neural network; LSTM: long short-term memory; GRU: gated recurrent unit.

DT had the highest F1 score when the information gain method and max depth were set to entropy and 9, respectively. RF had the highest F1 score when the information gain method and max depth were set to entropy and 9, respectively. SVM had the highest F1 score when the penalty of square l2 and kernel was set to 6.5 and radial basis function, respectively. AB had the highest F1 score when the number of estimators and learning rate were set to 150 and 1.0, respectively. GB had the highest F1 score when the number of estimators and max depth was set to 120 and 4, respectively. MLP had the highest F1 score when the unit size and dropout rate were set to 256 and 20%, respectively. RNN had the highest F1 score when the unit size and dropout rate were set to 128 and 30%, respectively. LSTM had the highest F1 score when the unit size and dropout rate were set to 128 and 40%, respectively. GRU had the highest F1 score when the unit size and dropout rate were set to 128 and 40%, respectively.

### *4.5. Evaluation of Oversampling Algorithms*

We separate our dataset into train and test datasets. In the case of shallow machine learning, we train algorithms using stratified k-folds by train folds after separating train datasets into train folds and holdout folds. We train deep learning using train folds and early stop using validation folds after separating train datasets into train, validation, and holdout folds. In addition, we compare the performance of each oversampling via each algorithm using holdout folds. Table 9 shows the performance of each machine learning algorithm and oversampling algorithm. Table 9 showed the best performance at GRU with ADASYN.

**Table 9.** Prediction performance. GRU with ADASYN demonstrated the highest performance.

| Machine Learning Algorithm | Oversampling | PPV | Sensitivity | F1 Score | AUC |
|---|---|---|---|---|---|
| LR | SMOTE | 42.42% | 86.39% | 0.5690 | 0.8817 |
| | Borderline SMOTE | 43.24% | 85.34% | 0.5739 | 0.8787 |
| | ADASYN | 44.24% | 86.39% | 0.5851 | 0.8853 |
| DT | SMOTE | 86.27% | 92.15% | 0.8911 | 0.9545 |
| | Borderline SMOTE | 92.22% | 93.19% | 0.9271 | 0.9626 |
| | ADASYN | 86.12% | 94.24% | 0.9000 | 0.9647 |
| RF | SMOTE | 96.43% | 98.95% | 0.9767 | 0.9932 |
| | Borderline SMOTE | 95.43% | 98.43% | 0.9691 | 0.9901 |
| | ADASYN | 96.43% | 98.90% | 0.9765 | 0.9929 |
| SVM | SMOTE | 91.09% | 96.34% | 0.9364 | 0.9776 |
| | Borderline SMOTE | 90.53% | 90.05% | 0.9029 | 0.9462 |
| | ADASYN | 90.40% | 93.72% | 0.9203 | 0.9643 |
| AB | SMOTE | 83.11% | 95.29% | 0.8878 | 0.9681 |
| | Borderline SMOTE | 85.57% | 90.05% | 0.8776 | 0.9438 |
| | ADASYN | 83.49% | 92.67% | 0.8784 | 0.9555 |
| GB | SMOTE | 95.96% | 99.48% | 0.9769 | 0.9956 |
| | Borderline SMOTE | 95.90% | 97.91% | 0.9689 | 0.9877 |
| | ADASYN | 98.45% | 99.48% | 0.9896 | 0.9967 |
| MLP | SMOTE | 98.94% | 97.91% | 0.9842 | 0.9891 |
| | Borderline SMOTE | 96.84% | 96.34% | 0.9659 | 0.9803 |
| | ADASYN | 98.38% | 95.29% | 0.9681 | 0.9758 |
| RNN | SMOTE | 97.33% | 95.29% | 0.9630 | 0.9753 |
| | Borderline SMOTE | 98.35% | 93.72% | 0.9598 | 0.9679 |
| | ADASYN | 96.35% | 96.86% | 0.9661 | 0.9827 |

**Table 9.** *Cont.*

| Machine Learning Algorithm | Oversampling | PPV | Sensitivity | F1 Score | AUC |
|---|---|---|---|---|---|
| LSTM | SMOTE | 97.85% | 95.29% | 0.9655 | 0.9755 |
| | Borderline SMOTE | 98.38% | 95.29% | 0.9681 | 0.9758 |
| | ADASYN | 98.93% | 96.86% | 0.9788 | 0.9838 |
| GRU | SMOTE | 98.3–8% | 95.29% | 0.9681 | 0.9758 |
| | Borderline SMOTE | 98.38% | 95.29% | 0.9681 | 0.9758 |
| | ADASYN | 98.42% | 97.91% | 0.9816 | 0.9889 |

PPV: positive predictive value; LR: logistic regression; DT: decision tree; RF: random forest; SVM: support vector machine; AB: adaptive boost; GB: gradient boost; MLP: multi-layer perceptron; RNN: recurrent neural network; LSTM: long short-term memory; GRU: gated recurrent unit.

### *4.6. Evaluation of Machine Learning Algorithms*

We separate our dataset into train and test datasets. In the case of shallow machine learning, we train algorithms using stratified k-folds by train folds. We train deep learning using train folds and early stop using validation folds after separating train datasets into train and validation folds. In addition, we compare the performance of each algorithm using test datasets. Table 10 shows the highest performance achieved for each machine learning algorithm. In the case of PPV, GRU shows the highest performance. In the case of sensitivity, GB shows the highest performance. In the case of the F1 score, LSTM shows the highest performance.

**Table 10.** Highest prediction performance of each machine learning algorithm.

| Machine Learning Algorithms | Oversampling | PPV | Sensitivity | F1 Score | AUC |
|---|---|---|---|---|---|
| LR | ADSYN | 44.47% | 83.88% | 0.5812 | 0.8745 |
| DT | Borderline SMOTE | 92.11% | 94.14% | 0.9312 | 0.9672 |
| RF | SMOTE | 96.09% | 98.90% | 0.9747 | 0.9928 |
| SVM | SMOTE | 89.49% | 96.70% | 0.9296 | 0.9786 |
| AB | SMOTE | 86.33% | 94.87% | 0.9040 | 0.9679 |
| GB | ADASYN | 96.10% | 99.27% | 0.9766 | 0.9946 |
| MLP | SMOTE | 97.48% | 99.27% | 0.9837 | 0.9952 |
| RNN | ADASYN | 95.67% | 97.07% | 0.9636 | 0.9835 |
| LSTM | ADASYN | 98.18% | 98.90% | 0.9854 | 0.9937 |
| GRU | ADASYN | 98.53% | 98.17% | 0.9835 | 0.9902 |

PPV: positive predictive value; LR: logistic regression; DT: decision tree; RF: random forest; SVM: support vector machine; AB: adaptive boost; GB: gradient boost; MLP: multi-layer perceptron; RNN: recurrent neural network; LSTM: long short-term memory; GRU: gated recurrent unit.

### *4.7. Comparison of the Importance of Disease Prediction*

We guess that the most important techniques in a machine learning algorithm are imputation, oversampling, and feature selection, in that order. We evaluate the performance of an RDI-based imputer and a mean-based imputer. In the RDI-based imputer, the PPV and sensitivity improved by more than 20% and 10%, respectively. The datasets in the medical field are unbalanced. In the case of diseases such as respiratory failure, they are classified according to their occurrence. The overfitting problem of prediction occurs when machine learning comprised a majority of the data in datasets. Oversampling or undersampling must be performed to solve imbalanced datasets. Table 11 shows the performance results of replacing missing values with the mean or not performing oversampling. In the case of not performing oversampling, the sensitivity is lower than if oversampling had been performed.

**Table 11.** Performance result of each scenario.

| Scenario | Algorithm | PPV | Sensitivity | F1 Score | AUC |
|---|---|---|---|---|---|
| Reference | LR | 44.47% | 83.88% | 0.5812 | 0.8745 |
| | DT | 92.11% | 94.14% | 0.9312 | 0.9672 |
| | RF | 96.09% | 98.90% | 0.9747 | 0.9928 |
| | SVM | 89.49% | 96.70% | 0.9296 | 0.9786 |
| | AB | 86.33% | 94.87% | 0.9040 | 0.9679 |
| | GB | 96.10% | 99.27% | 0.9766 | 0.9946 |
| | MLP | 97.48% | 99.27% | 0.9837 | 0.9952 |
| | RNN | 95.67% | 97.07% | 0.9636 | 0.9835 |
| | LSTM | 98.18% | 98.90% | 0.9854 | 0.9937 |
| | GRU | 98.53% | 98.17% | 0.9835 | 0.9902 |
| Replace missing values via average | LR | 22.26% | 70.70% | 0.3386 | 0.7477 |
| | DT | 63.06% | 83.15% | 0.7172 | 0.8949 |
| | RF | 53.24% | 87.18% | 0.6611 | 0.9031 |
| | SVM | 41.05% | 83.15% | 0.5496 | 0.8646 |
| | AB | 58.21% | 73.99% | 0.6516 | 0.8472 |
| | GB | 75.40% | 86.45% | 0.8055 | 0.9201 |
| | MLP | 65.49% | 81.32% | 0.7255 | 0.8882 |
| | RNN | 62.10% | 78.02% | 0.6916 | 0.8697 |
| | LSTM | 41.63% | 80.22% | 0.5482 | 0.8529 |
| | GRU | 65.73% | 85.71% | 0.7440 | 0.9094 |
| Does not perform oversampling | LR | 81.46% | 45.05% | 0.5802 | 0.7209 |
| | DT | 98.05% | 92.31% | 0.9509 | 0.9608 |
| | RF | 98.85% | 94.14% | 0.9644 | 0.9702 |
| | SVM | 99.06% | 77.29% | 0.8683 | 0.8861 |
| | AB | 94.80% | 86.81% | 0.9063 | 0.9320 |
| | GB | 98.38% | 89.01% | 0.9346 | 0.9444 |
| | MLP | 95.67% | 80.95% | 0.8770 | 0.9032 |
| | RNN | 90.87% | 83.88% | 0.8724 | 0.9158 |
| | LSTM | 77.73% | 62.64% | 0.6937 | 0.8055 |
| | GRU | 91.09% | 86.08% | 0.8851 | 0.9268 |

PPV: positive predictive value; LR: logistic regression; DT: decision tree; RF: random forest; SVM: support vector machine; AB: adaptive boost; GB: gradient boost; MLP: multi-layer perceptron; RNN: recurrent neural network; LSTM: long short-term memory; GRU: gated recurrent unit.

Table 4 shows the variance in performance according to the features. We consider the features with the highest performance in Table 4. In the case of machine learning algorithms, the performance depends on the dataset. The sensitivity of machine learning algorithms in Table 10 is more than 80%. However, the PPV of logistic regression is less than 50%. The deep learning models such as RNN, LSTM, and GRU all performed at comparable levels.

## 5. Discussion

We proposed the prediction model for acute respiratory failure, an important prognostic factor in acute pesticide poisoning patients. The effects of respiratory failure include loss of consciousness, arrhythmias, and death. Table 12 shows the performance of each

algorithm for the prediction of respiratory failure. In recent years, respiratory failure prediction models have been developed to predict respiratory failure in COVID-19 patients based on deep learning with semi-supervised learning [4], respiratory failure based on XGBoost using clinical data [5], respiratory failure in COVID-19 patients based on LR [6], respiratory failure in ICU patients based on LightGBM [7], respiratory failure in patients with pesticide poisoning due to intentional pesticide ingestion based on LR [9], cardiac arrest and respiratory failure in ICU patients based on LSTM [10], and respiratory failure in ICU patients based on gradient boosting [8]. In the case of prediction based on semi-supervised learning [4], the PPV was 0.033 and the sensitivity was 0.78. In the case of prediction based on XGBoost using clinical data [5], the sensitivity was 0.71. In the case of prediction of respiratory failure with COVID-19 based on LR [6], the PPV was 0.74 and the sensitivity was 0.72. In the case of prediction based on LightGBM [7], the PPV was 0.42 and the sensitivity was 0.80. In the case of prediction of respiratory failure in patients with pesticide poisoning based on LR [9], the PPV was 0.833 and the sensitivity was 0.606. In the case of prediction based on LSTM [10], the PPV was 0.226 and the sensitivity was 0.881. However, these respiratory failure prediction algorithms are characterized by a large measurement interval, low performance, or large number of features [4–10]. Our proposed algorithm demonstrated improved respiratory failure prediction within 24 h with higher PPV and sensitivity compared with those of other models.

**Table 12.** Comparison of the performance between the proposed algorithm and algorithms in other studies.

| | Algorithms | Features | Patient Data Range | Sensitivity | PPV | AUC |
|---|---|---|---|---|---|---|
| [4] | Semi-supervised learning | 25 | 32 h | 0.78 | 0.023 | 0.78 |
| [5] | XGBoost | 24 | - | 0.71 | - | - |
| [6] | LR | 26 | - | 0.72 | 0.74 | 0.89 |
| [7] | LightGBM | 25 | - | 0.80 | - | 0.746 |
| [8] | GradientBoosting | 106 | 6 h | 0.534 | 0.643 | 0.769 |
| [9] | LR | 7 | - | 0.606 | 0.833 | 0.912 |
| [10] | LSTM | 8 | 2 h | 0.881 | 0.226 | 0.886 |
| Our algorithm | LSTM | 17 | 3 h | 0.9817 | 0.9890 | 0.9937 |

PPV: positive predictive value; LR: logistic regression; LSTM: long short-term memory; RF: random forest.

Our proposed algorithm demonstrated improved respiratory failure prediction within 24 h with higher PPV and sensitivity compared with those of other models. We guess that the pesticide category, white blood cell (WBC), pH, heart rate (HR), and C-reactive protein 1 are important predictors of respiratory failure in acute pesticide poisoning. The performance results of important features based on RF and GB confirmed that the highest-scoring features are the above features. These are the limitations of this paper: First, we conducted a single cohort at the Soonchunhyang University Cheonan Hospital. Second, we proceeded with retrospective research. We have not yet confirmed the validity from an external source. Third, our algorithm required three-hour EMRs. Our algorithm is not applicable to high-risk patients hospitalized for less than three hours. Fourth, our algorithm is difficult to use in hospitals. Our algorithm predicts whether respiratory failure has happened within 24 h. It does not estimate the time or risk score that a patient should experience respiratory failure. Follow-up research is required to decrease the prediction range for respiratory failure or score the patient's risk or provide information such as the estimated time of respiratory failure.

## 6. Conclusions

We predicted respiratory failure in patients with pesticide poisoning at Soonchunhyang University Cheonan Hospital. We analyzed the 3 h medical records of individuals with pesticide poisoning to predict respiratory failure within 24 h. In consideration of time-series properties, sliding windows, feature selection, and oversampling were used to replace missing values. Enhanced performance was achieved with the use of LSTM. Moreover, our machine learning technique algorithm could improve the prognosis of patients with pesticide poisoning. In addition, we will enhance the algorithm for predicting respiratory failure within 24 h such that it can predict respiratory failure within 4 or 8 h. We plan to conduct studies to predict respiratory failure in patients admitted to the general ward and ICU.

**Author Contributions:** Conceptualization, H.L. and H.G.; methodology, Y.K. and M.C.; software, Y.K. and M.C.; validation, N.C., H.G. and H.L.; formal analysis, Y.K.; resources, H.G. and N.C.; data curation, N.C.; writing—original draft preparation, Y.K. and M.C.; writing—review and editing, H.L. and H.G.; visualization, Y.K. and M.C.; supervision, H.L.; project administration, H.L. and H.G.; funding acquisition, H.L. and H.G. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ICAN (ICT Challenge and Advanced Network of HRD) program (IITP-2022-RS-2022-00156439) supervised by the IITP (Institute of Information and Communications Technology Planning and Evaluation), the Bio and Medical Technology Development Program (No. NRF-2019M3E5D1A02069073) and a Korea University Grant.

**Institutional Review Board Statement:** The study was conducted in accordance with the Declaration of Helsinki and approved by the Institutional Review Board of Soonchunhyang University Cheonan Hospital (IRB number: 2020-02-016).

**Informed Consent Statement:** Patient consent was waived because of the retrospective design of the study.

**Data Availability Statement:** Data sharing not applicable.

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

## References

1. Cho, N.-J.; Park, S.; Lee, E.Y.; Gil, H.-W. Risk factors to predict acute respiratory failure in patients with acute pesticide poisoning. *J. Korean Soc. Clin. Toxicol.* **2020**, *18*, 116–122.
2. Lee, H.; Choa, M.; Han, E.; Ko, D.R.; Ko, J.; Kong, T.; Cho, J.; Chung, S.P. Causative Substance and Time of Mortality Presented to Emergency Department Following Acute Poisoning: 2014-2018 National Emergency Department Information System (NEDIS). *J. Korean Soc. Clin. Toxicol.* **2021**, *19*, 65–71.
3. Mew, E.J.; Padmanathan, P.; Konradsen, F.; Eddleston, M.; Chang, S.-S.; Phillips, M.R.; Gunnell, D. The global burden of fatal self-poisoning with pesticides 2006-15: Systematic review. *J. Affect. Disord.* **2017**, *219*, 93–104. [CrossRef] [PubMed]
4. Lam, C.; Tso, C.F.; Green-Saxena, A.; Pellegrini, E.; Iqbal, Z.; Evans, D.; Hoffman, J.; Calvert, J.; Mao, Q.; Das, R. Semisupervised Deep Learning Techniques for Predicting Acute Respiratory Distress Syndrome from Time-Series Clinical Data: Model Development and Validation Study. *JMIR Form. Res.* **2021**, *5*, e28028. [CrossRef]
5. Sinha, P.; Churpek, M.M.; Calfee, C.S. Machine learning classifier models can identify acute respiratory distress syndrome phenotypes using readily available clinical data. *Am. J. Respir. Crit. Care Med.* **2020**, *202*, 996–1004. [CrossRef]
6. Bartoletti, M.; Giannella, M.; Scudeller, L.; Tedeschi, S.; Rinaldi, M.; Bussini, L.; Fornaro, G.; Pascale, R.; Pancaldi, L.; Pasquini, Z. Development and validation of a prediction model for severe respiratory failure in hospitalized patients with SARS-CoV-2 infection: A multicentre cohort study (PREDI-CO study). *Clin. Microbiol. Infect.* **2020**, *26*, 1545–1553. [CrossRef]
7. Hüser, M.; Faltys, M.; Lyu, X.; Barber, C.; Hyland, S.L.; Merz, T.M.; Rätsch, G. Early prediction of respiratory failure in the intensive care unit. *arXiv* **2021**, arXiv:2105.05728.
8. Schwager, E.; Jansson, K.; Rahman, A.; Schiffer, S.; Chang, Y.; Boverman, G.; Gross, B.; Xu-Wilson, M.; Boehme, P.; Truebel, H. Utilizing machine learning to improve clinical trial design for acute respiratory distress syndrome. *NPJ Digit. Med.* **2021**, *4*, 133. [CrossRef]
9. Cho, N.-J.; Park, S.; Lyu, J.; Lee, H.; Hong, M.; Lee, E.-Y.; Gil, H.-W. Prediction Model of Acute Respiratory Failure in Patients with Acute Pesticide Poisoning by Intentional Ingestion: Prediction of Respiratory Failure in Pesticide Intoxication (PREP) Scores in Cohort Study. *J. Clin. Med.* **2022**, *11*, 1048. [CrossRef] [PubMed]

10. Kim, J.; Chae, M.; Chang, H.-J.; Kim, Y.-A.; Park, E. Predicting cardiac arrest and respiratory failure using feasible artificial intelligence with simple trajectories of patient data. *J. Clin. Med.* **2019**, *8*, 1336. [CrossRef]
11. Idri, A.; Benhar, H.; Fernández-Alemán, J.; Kadi, I. A systematic map of medical data preprocessing in knowledge discovery. *Comput. Methods Programs Biomed.* **2018**, *162*, 69–85. [CrossRef]
12. Benhar, H.; Idri, A.; Fernández-Alemán, J. Data preprocessing for heart disease classification: A systematic literature review. *Comput. Methods Programs Biomed.* **2020**, *195*, 105635. [CrossRef] [PubMed]
13. Jadhav, A.; Pramod, D.; Ramanathan, K. Comparison of performance of data imputation methods for numeric dataset. *Appl. Artif. Intell.* **2019**, *33*, 913–933. [CrossRef]
14. Li, J.; Cheng, K.; Wang, S.; Morstatter, F.; Trevino, R.P.; Tang, J.; Liu, H. Feature selection: A data perspective. *ACM Comput. Surv.* **2017**, *50*, 1–45. [CrossRef]
15. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Du-bourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res. JMLR* **2011**, *12*, 2825–2830.
16. Stekhoven, D.J.; Bühlmann, P. MissForest—Non-parametric missing value imputation for mixed-type data. *Bioinformatics* **2012**, *28*, 112–118. [CrossRef]
17. Seabold, S.; Perktold, J. Statsmodels: Econometric and statistical modeling with python. In Proceedings of the 9th Python in Science Conference, Austin, TX, USA, 28 June–3 July 2010; p. 10-25080.
18. Mohammed, R.; Rawashdeh, J.; Abdullah, M. Machine learning with oversampling and undersampling techniques: Over-view study and experimental results. In Proceedings of the 2020 11th International Conference on Information and Communication Systems (ICICS), Irbid, Jordan, 7–9 April 2020; pp. 243–248.
19. He, H.; Bai, Y.; Garcia, E.A.; Li, S. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In Proceedings of the 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Hong Kong, China, 1–8 June 2008; pp. 1322–1328.
20. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
21. Han, H.; Wang, W.-Y.; Mao, B.-H. Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In Proceedings of the International Conference on Intelligent Computing, Hefei, China, 23–26 August 2005; pp. 878–887.
22. Kleinbaum, D.G.; Dietz, K.; Gail, M.; Klein, M.; Klein, M. *Logistic regression*; Springer: Berlin/Heidelberg, Germany, 2002.
23. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
24. Platt, J. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Adv. Large Margin Classif.* **1999**, *10*, 61–74.
25. Freund, Y.; Schapire, R.E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* **1997**, *55*, 119–139. [CrossRef]
26. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A System for Large-Scale Machine Learning. In Proceedings of the 12th USENIX symposium on operating systems design and implementation (OSDI 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
27. Hüsken, M.; Stagge, P. Recurrent neural networks for time series classification. *Neurocomputing* **2003**, *50*, 223–235. [CrossRef]
28. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
29. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
30. Refaeilzadeh, P.; Tang, L.; Liu, H. Cross-validation. *Encycl. Database Syst.* **2009**, *5*, 532–538.
31. Berrar, D. Cross-Validation. *Encycl. Bioinform. Comput. Biol.* **2019**, *1*, 542–545. Available online: https://www.sciencedirect.com/science/article/pii/B978012809633820349X?via%3Dihub (accessed on 23 September 2022).