algorithms

MDPI

Article

COVID-19 Prediction Applying Supervised Machine Learning Algorithms with Comparative Analysis Using WEKA

Charlyn Nayve Villavicencio [1,2,*], Julio Jerison Escudero Macrohon [1], Xavier Alphonse Inbaraj [1], Jyh-Horng Jeng [1] and Jer-Guang Hsieh [3]

[1] Department of Information Engineering, I-Shou University, Kaohsiung City 84001, Taiwan; isu10903050D@cloud.isu.edu.tw (J.J.E.M.); xalphonse@gmail.com (X.A.I.); jjeng@isu.edu.tw (J.-H.J.)
[2] College of Information and Communications Technology, Bulacan State University, Bulacan 3000, Philippines
[3] Department of Electrical Engineering, I-Shou University, Kaohsiung City 84001, Taiwan; jghsieh@gmail.com
* Correspondence: charlyn.villavicencio@bulsu.edu.ph; Tel.: +886-95-845-0028

Abstract: Early diagnosis is crucial to prevent the development of a disease that may cause danger to human lives. COVID-19, which is a contagious disease that has mutated into several variants, has become a global pandemic that demands to be diagnosed as soon as possible. With the use of technology, available information concerning COVID-19 increases each day, and extracting useful information from massive data can be done through data mining. In this study, authors utilized several supervised machine learning algorithms in building a model to analyze and predict the presence of COVID-19 using the COVID-19 Symptoms and Presence dataset from Kaggle. J48 Decision Tree, Random Forest, Support Vector Machine, K-Nearest Neighbors and Naïve Bayes algorithms were applied through WEKA machine learning software. Each model's performance was evaluated using 10-fold cross validation and compared according to major accuracy measures, correctly or incorrectly classified instances, kappa, mean absolute error, and time taken to build the model. The results show that Support Vector Machine using Pearson VII universal kernel outweighs other algorithms by attaining 98.81% accuracy and a mean absolute error of 0.012.

Keywords: algorithms; COVID-19; COVID-19 symptoms; classification; data mining; health care; supervised machine learning; waikato environment for knowledge analysis; WEKA

Citation: Villavicencio, C.N.; Macrohon, J.J.E.; Inbaraj, X.A.; Jeng, J.-H.; Hsieh, J.-G. COVID-19 Prediction Applying Supervised Machine Learning Algorithms with Comparative Analysis Using WEKA. *Algorithms* **2021**, *14*, 201. https://doi.org/10.3390/a14070201

Academic Editor: Frank Werner

Received: 30 May 2021
Accepted: 28 June 2021
Published: 30 June 2021

1. Introduction

Coronavirus Disease (COVID-19) is an infectious disease caused by a novel coronavirus that originated in Wuhan China last December (2019). This disease will affect the respiratory system of a person, and some people will eventually get better without having special treatment, especially for those who have a strong immune system [1]. For others, though, it may be different—old persons are more vulnerable, including those with existing comorbidities such as cardiovascular disease, diabetes, respiratory disease and cancer. COVID-19 is not just a respiratory disease, it is multisystemic. Recent studies determined that this virus affects almost all the organs of the body, which is stimulated by its widespread inflammatory response [2]. Moreover, about 10–15% of COVID-19 patients develop severe symptoms; these individuals may experience long COVID-19, which may cause complications to the heart, lungs, and nervous system [3]. COVID-19 can spread because this virus is transmissible by droplets into the air from the infected person through speaking, coughing, and sneezing, or even touching some contaminated objects or areas. The World Health Organization (WHO) stated that frequent handwashing, disinfecting, social distancing, wearing masks and not touching your face can protect one from being infected. The WHO listed several symptoms and emphasized that fever, a dry cough, and tiredness were the most common, while less common symptoms were headaches, sore throat, diarrhea, conjunctivitis, loss of smell, and rashes, and serious symptoms were breathing problems, chest pain and loss of speech and movement. As of 29 June 2021, there

were 182,333,954 COVID-19 cases and 3,948,238 deaths worldwide [4], and this disease had mutated into several variants documented in countries such as the United Kingdom, South Africa, the United States, India and Brazil, which brings increased severity to the disease, as well as quicker transmission, a higher death rate and reduced effectivity of vaccines [5]. As the virus keeps on spreading despite the efforts of the community to contain the virus, an outbreak can lead to increased demands in hospital resources, and shortages of medical equipment, healthcare staff and of course COVID-19 testing kits [6]. Limited access to COVID-19 testing kits can hinder the early diagnosis of the disease, and giving the best possible care for the suspected COVID-19 patients can be burdensome. Consequently, an automatic prediction system that aims to determine the presence of COVID-19 in a person is essential. Machine learning classification algorithms, datasets and machine learning software are the necessary tools for designing a COVID-19 prediction model.

Machine learning can be categorized as supervised, unsupervised, and reinforcement learning. Supervised machine learning is an approach that trains the machine using labeled datasets, wherein the examples are correctly labeled according to the class to which they belong [7]. The machine will analyze the given data and will eventually predict new instances based on information learned from the past data. Unlike supervised machine learning, the unsupervised machine learning learns by itself without the presence of the correctly labeled data. In unsupervised machine learning, the machine will be fed by the training samples, and it is the job of the machine to determine the hidden patterns from the dataset. For the reinforcement learning, the machine acts as an agent that aims to discover the most appropriate actions through a trial-and-error approach and observation in the environment [8]. Every time the machine successfully performs a task, it will be rewarded by increasing its state; otherwise, it will be punished by decreasing its state, and this approach will be repeated several times until the machine learns how to perform a specific task correctly. Reinforcement learning is used in training robots in how to perform human-like tasks and personal assistance.

This study is mainly focused on predicting the presence of COVID-19 in a person; thus, a supervised machine learning model had to be developed. Several machine learning methods were utilized in building disease prediction models (e.g., coronary artery disease, respiratory disease, breast cancer, diabetes, dementia, and fatty liver disease [9–14]). The researchers devised a list of published disease prediction studies that utilized supervised machine learning algorithms, such as J48 Decision Tree (J48 DT), Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbors (k-NN), and Naïve Bayes (NB). More algorithms were utilized, such as Multi-layer Perceptron (MLP), Logistic Regression (LR) and Artificial Neural Network (ANN). Table 1 displays a list of disease prediction studies, machine learning algorithms utilized, the best algorithms according to the experiments performed, and the accuracy obtained.

Table 1. Machine learning algorithms used in building a disease prediction model; the best algorithm and the accuracy obtained are displayed.

Reference	Disease Prediction	Machine Learning Algorithms	Best Algorithm	Accuracy Obtained
[9]	Coronary Artery Disease	LR, DT, RF, SVM, k-NN, ANN	ANN	93.03%
[10]	Respiratory Disease	RF, DT, MLP	DT	99.41%
[11]	Breast Cancer	SVM, DT, NB, and k-NN	SVM	97.13%
[12]	Diabetes	DT, SVM, and NB	NB	76.30%
[13]	Dementia	J48 DT, NB, RF, and MLP	J48 DT	99.23%
[14]	Fatty Liver Disease	LR, k-NN, DT, SVM, NB, and RF	LR	75.00%
Proposed Method	COVID-19	J48 DT, NB, SVM, k-NN, and RF	SVM	98.81%

Several supervised machine learning methods were utilized in building a disease prediction model, and it is apparent that an algorithm can perform differently depending on the dataset. There were several research attempts in using machine learning for the early prediction of this novel disease. Based on the laboratory findings, a COVID-19 predictor has

been developed using Convolutional Neural Network (CNN); the researchers developed this predictor using the dataset from a hospital in Sao Paul, Brazil, and this model yielded 76% accuracy [15]. Another research attempt to improve COVID-19 severity prediction was undertaken using initial test results from the blood routine, which disclosed that age, white blood cells, lymphocytes, and neutrophils were the key factors in severity, and concluded that it could successfully improve the prediction accuracy of the model [16]. To accelerate the prediction of COVID-19, publicly available X-ray images were also used for training, and the best-performing model has been integrated into smart phones and evaluated. It was concluded that VGG16 generated a higher positive likelihood and lower negative likelihood in terms of predicting COVID-19 [17]. Moreover, another study used machine learning in predicting the mortality of a COVID-19 patient through the development of an XGBoost model, which successfully predicted the mortality of the patients earlier by about 10 days, having an accuracy of more than 90% [18].

Machine learning was also utilized in the development of models that can forecast cases of COVID-19. Using an algorithm called Stochastic Fractal search with a mathematical model, the number of life-threatening cases, deaths, and recovered, symptomatic and asymptomatic patients were predicted [19]. Predicting confirmed and death cases daily on the Asian continent was also made possible by using the Polynomial Neural Network method [20]. Using a prediction model in the study of Caballero et al. [21], the overall infection count and the days on which there will be a possible COVID-19 outbreak in Italy and Spain can be predicted one week before occurrence. These are some examples of studies that applied machine learning in predicting COVID-19 presence, cases, and death rates. However, there is a lack of existing literatures that utilize several supervised machine learning algorithms in developing a COVID-19 predictor using symptoms that the person is experiencing as the parameters. This study aims to build a model that can automatically predict the presence of COVID-19 in a person utilizing J48 DT, RF, SVM, k-NN, and NB algorithms, by analyzing COVID-19 symptoms using Waikato Environment for Knowledge Analysis (WEKA), which is an open-source software developed at the University of Waikato, New Zealand. WEKA is a collection of machine learning algorithms that supports data mining tasks by providing a wide range of tools that can be used for data pre-processing, classification, clustering, regression, association and visualization [22]. The name "Weka" was derived from a flightless bird found in the islands of New Zealand. The authors utilized two applications of WEKA named "Explorer" and "Knowledge Flow" in building a COVID-19 prediction model. Data pre-processing, classification and visualization processes were also performed using WEKA machine learning software.

2. Materials and Methods

This study was conducted using the block diagram shown in Figure 1.

2.1. Data Collection

For the data collection, the researchers used a dataset available from Kaggle entitled "COVID-19 Symptoms and Presence". This dataset has 20 attributes that are possible factors related to acquiring the virus, and 1 class attribute that determines the presence of COVID-19. Table 2 displays the features and descriptions of the dataset.

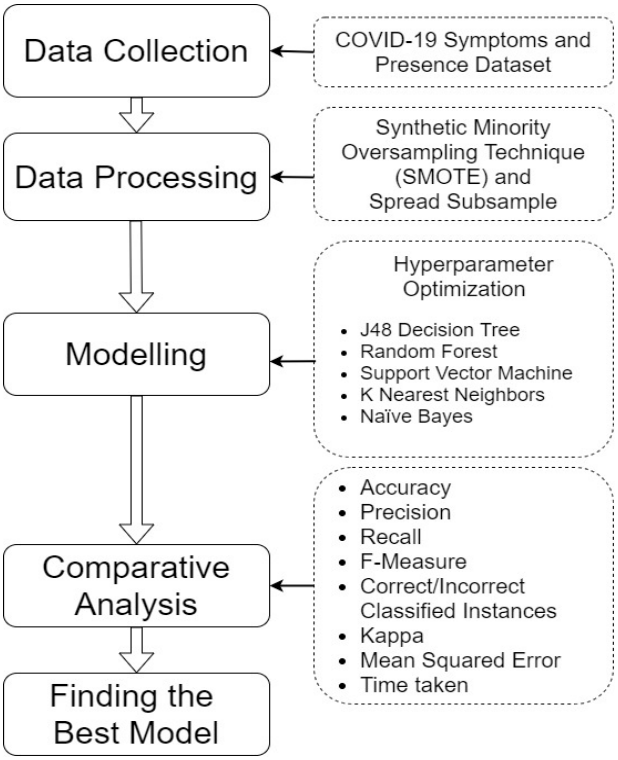


Figure 1. The five phases of this study are data collection, preprocessing, modelling, comparative analysis and finding the best model to determine COVID-19 presence.

Table 2. Features, percentage level and attribute descriptions of the COVID-19 symptoms and presence dataset.

Attribute Name	Type	Percentage Level	Description
Breathing Problem	Nominal	10%	The person is experiencing shortness of breath.
Fever	Nominal	10%	Temperature is above normal.
Dry Cough	Nominal	10%	Continuous coughing without phlegm.
Sore Throat	Nominal	10%	The person is experiencing sore throat.
Running Nose	Nominal	5%	The person is experiencing a runny nose.
Asthma	Nominal	4%	The person has asthma.
Chronic Lung Disease	Nominal	6%	The person has lung disease.
Headache	Nominal	4%	The person is experiencing headache.
Heart Disease	Nominal	2%	The person has cardiovascular disease.
Diabetes	Nominal	1%	The person is suffering from or has a history of diabetes.
Hypertension	Nominal	1%	Having a high blood pressure.
Fatigue	Nominal	2%	The person is experiencing tiredness.
Gastrointestinal	Nominal	1%	Having some gastrointestinal problems.
Abroad Travel	Nominal	8%	Recently went out of the country.
Contact with COVID-19 Patient	Nominal	8%	Had some close contact with people infected with COVID-19.
Attended Large Gathering	Nominal	6%	The person or anyone from their family recently attended a mass gathering.
Visited Public Exposed Places	Nominal	4%	Recently visited malls, temples, and other public places.
Family Working in Public Exposed Places	Nominal	4%	The person or anyone in their family is working in a market, hospital, or another crowded place.
Wearing Masks	Nominal	2%	The person is wearing face masks properly.
Sanitation from Market	Nominal	2%	Sanitizing products bought from market before use.
COVID-19	Nominal	-	The presence of COVID-19.

This dataset is visible to the public; its sources are the World Health Organization (WHO) Coronavirus Symptoms and the All India Institute of Medical Sciences (AIIMS).

The spatial coverage of this dataset is worldwide, and the date coverage is from 17 April 2020 to 29 August 2020 [23].

2.2. Data Processing

To process the dataset, the authors used the WEKA machine learning software's Explorer and Knowledge Flow applications, and the main page of WEKA is presented in Figure 2.

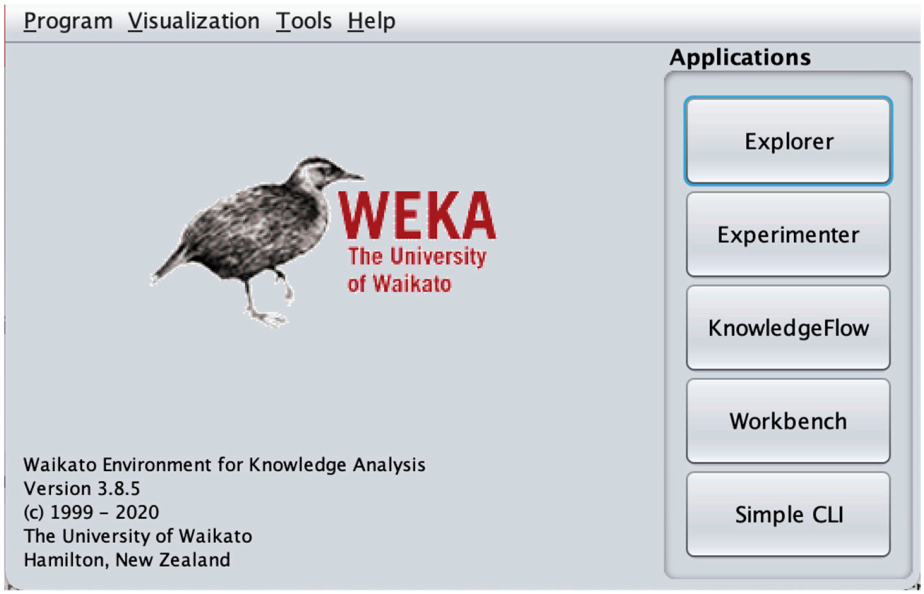


Figure 2. The main page of WEKA displaying the five modules including Explorer, Experimenter, Knowledge Flow, Workbench and Simple CLI.

There are various document formats accepted in WEKA, such as arff, C4.5, csv, JSON file, etc., and the COVID-19 Symptoms and Presence dataset is in csv format, which makes it easy to be imported and analyzed in the software. Data preprocessing can be started by clicking the open file and browsing for the location of the dataset. Once the dataset is imported in WEKA, the current relation, attributes, instances and the sum of weights and visualizations pertaining to the dataset will be displayed. The Preprocess tab in the WEKA Explorer application is displayed in Figure 3.

Once an attribute is selected, values such as name; missing, distinct and unique values; data type; labels; count, and weights will be visible. Distribution between the class labels is also displayed in a bar chart by clicking the COVID-19 attribute seen on the left side of the form. For the COVID-19 Symptoms and Presence dataset, the classes have a 4:1 class imbalance. The researchers used the synthetic minority oversampling technique (SMOTE) to generate additional instances for the minority group, which is the class labeled "No". SMOTE oversamples the minority class by generating additional synthetic samples, and in this way, the class with fewer samples will be increased. Combining the method of oversampling the minority class and undersampling the majority class, or cutting off some samples in the class containing more samples, will yield better classifier performance than just undersampling the majority class [24]. The SMOTE can be used in WEKA by installing it using the Package Manager under the Tools menu of the WEKA GUI Chooser. After installation, the package can be found by opening the Filters folder, then the Supervised folder, and the Instance folder. The details of the updated dataset can be seen in Figure 4.

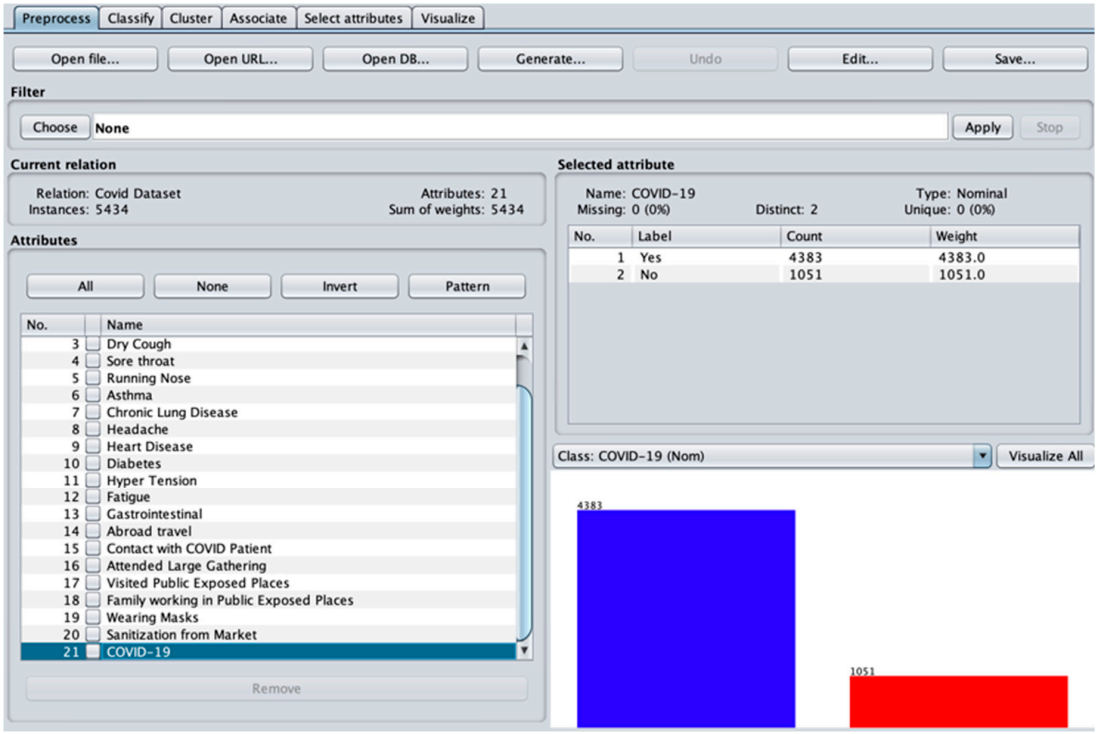


Figure 3. The Preprocess tab of WEKA Explorer showing the composition of the imported dataset and some visualizations. Users can click the Visualize All button to become more familiarized with the dataset.

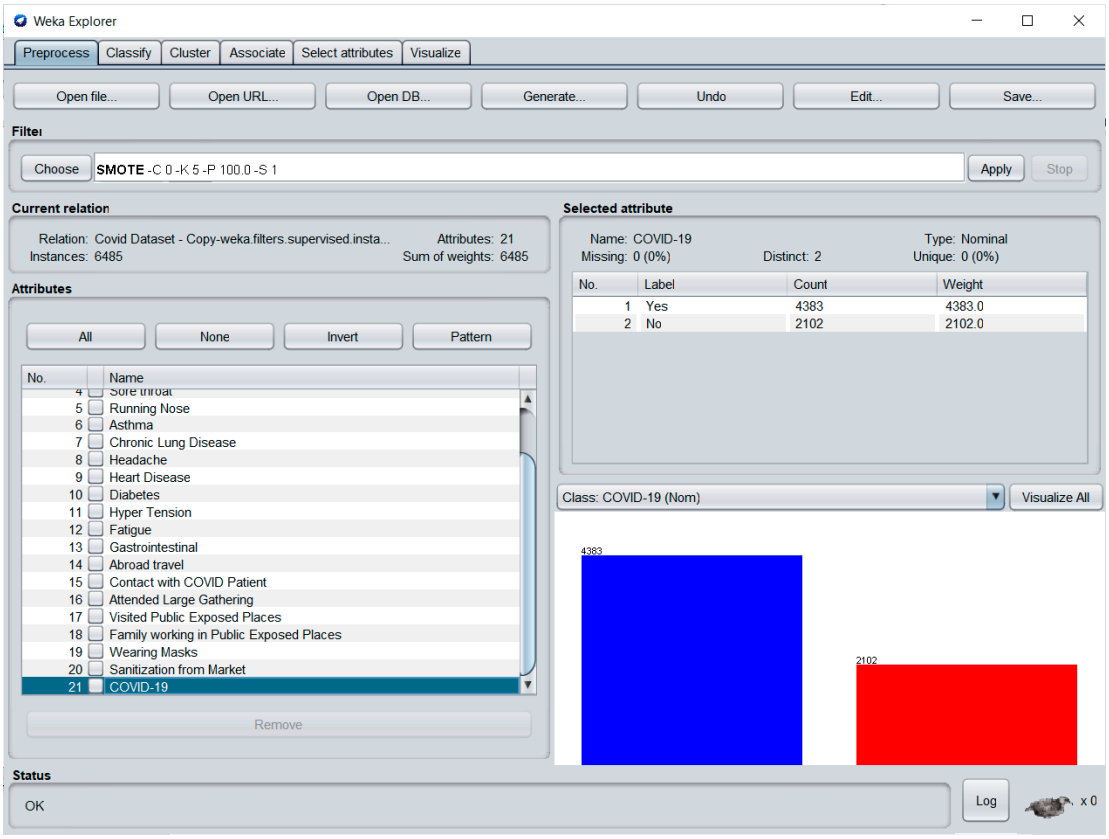


Figure 4. The Preprocess tab of WEKA Explorer showing the result of balancing the dataset through SMOTE using a bar graph for representing the number of instances per class.

In Figure 4, it can be seen that the red bar has been increased, indicating that there are additional samples added into the "No" class. Now that the minority class has gained its number, the dataset is still imbalanced. To address this, the researchers used the Spread Subsample to reduce the number in the majority class and make it even with the minority class. The updated numbers of the classes in the dataset can be seen in Figure 5.

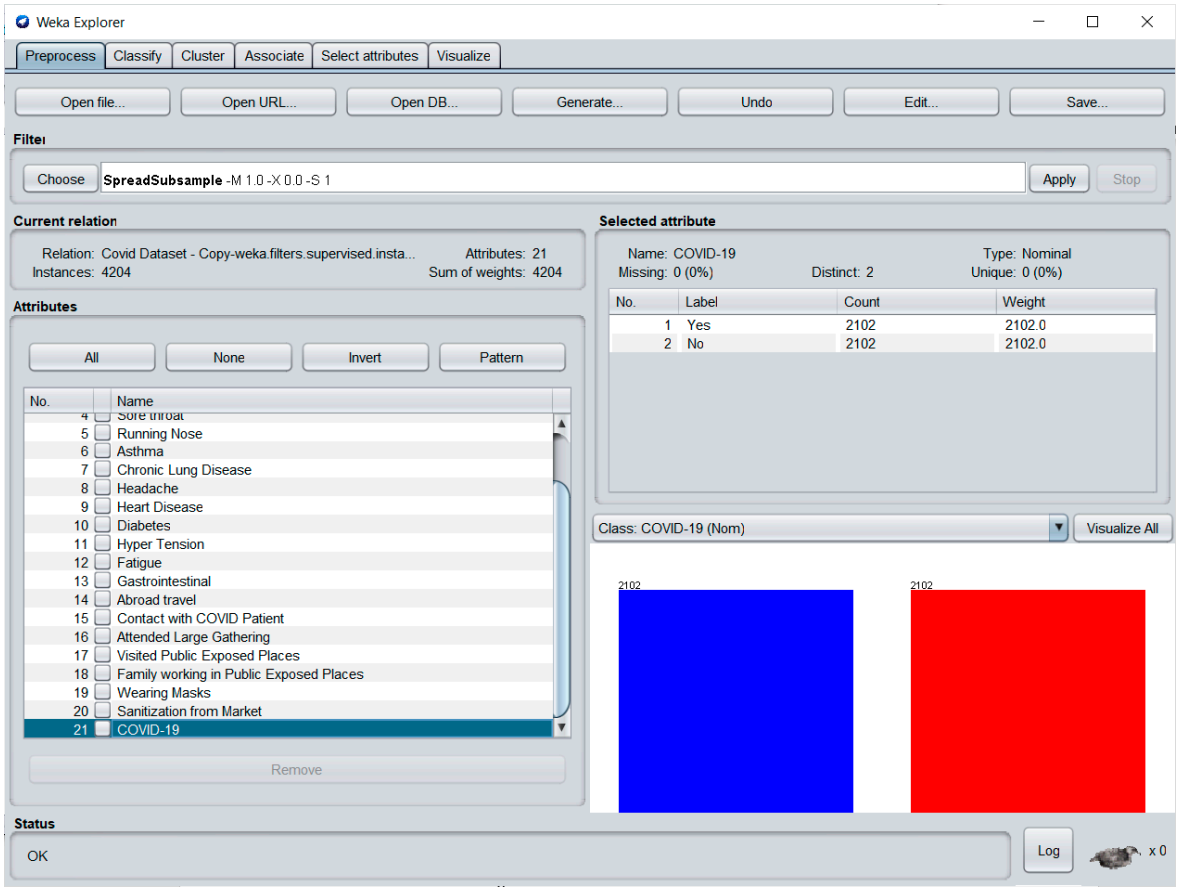


Figure 5. The Preprocess tab of WEKA Explorer showing the result of balancing the dataset through Spread Subsample or undersampling the majority class to make it equal to the minority class.

In Figure 5, it can be noted that the heights of the blue and red bars are now equal, indicating that both classes have the same values. This is the dataset that has been utilized in this study after implementing both the SMOTE and the Spread Subsample techniques. It is important to use a balanced dataset so that the classifier is well informed on both classes to be predicted, as well as to avoid distribution bias [25].

2.3. Modelling

After data processing using the SMOTE and Spread Subsample techniques, several models were built using the WEKA Explorer module utilizing different supervised machine learning algorithms, namely, J48 DT, RF, SVM, k-NN, and NB. In the Classify tab of the WEKA Explorer, the researchers chose the classifier name, and selected 10-fold cross-validation. To determine the best configuration for each algorithm, the researchers performed hyperparameter optimization by performing several trainings with the same dataset for each algorithm. The researchers used 10-fold cross-validation testing and a batch size of 100 for all the experiments.

For the J48 DT, the researchers used two as the minimum number of instances per leaf for all the training processes. The researchers also used a confidence factor for pruning, and unpruned either true or false as the parameters to be tuned. For the RF, the researchers set

the maximum depth to zero, which means unlimited depth, and the number of iterations to 100, which is the number of trees in the forest. Bagsize is the parameter that the researchers tuned to determine the best bag size that will yield good results. For the SVM, the C and Kernel hyperparameters were used. The C is the complexity parameter, which determines how flexible the process is in determining the line that separates the classes, and the Kernel determines how the data will be separated, either by a straight line, a curved line, a polygonal or a shape similar to that used in [26]. In building the model using the k-NN algorithm, the distance function used was Euclidean distance, and the hyperparameters tuned were the KNN or the number of neighbors to use and the Cross-Validate parameter, which indicates whether a cross-validation will be used in determining the best k value. Lastly, for the NB algorithm, the hyperparameters used were whether the process would use Kernel Estimator and Supervised Discretization rather than the normal distribution of numeric attributes.

Each algorithm underwent several experimental trainings using the WEKA Explorer Classify tab. For each training, the performance of the developed model was displayed in the classifier output section. An example of the classifier tab with the details of the developed models is displayed in Figure 6.

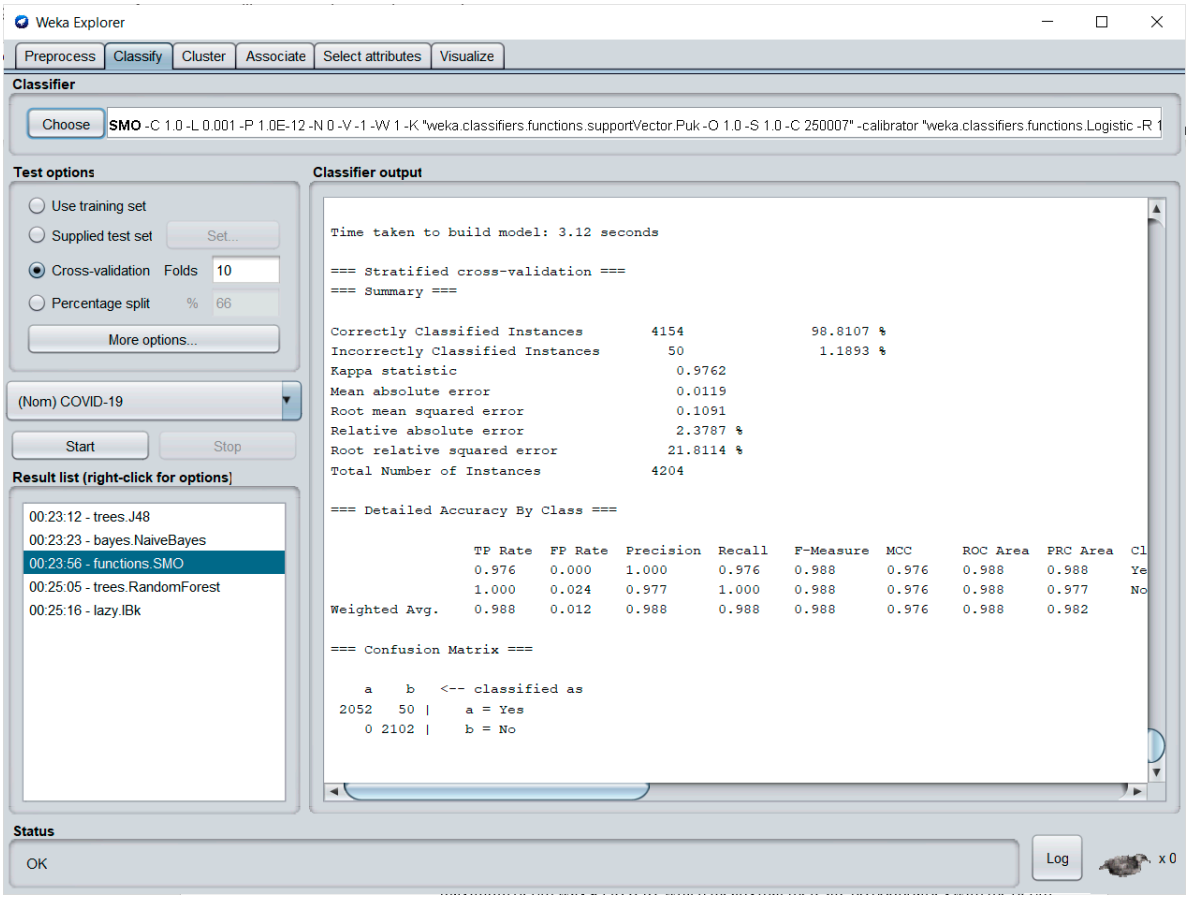


Figure 6. The Classify tab of WEKA Explorer wherein the user can choose different algorithms to be applied to the dataset. The details concerning the developed model's performance were displayed in the classifier output section.

In Figure 6, the Classify tab is displayed. When the user clicks the choose button, the available machine learning algorithm classifiers will be displayed. Several folders with a list of algorithms to be used will be presented, and both J48 DT and RF can be found under the trees folder, while NB is under the Bayes folder, SVM belongs to the functions folder and k-NN can be seen in the lazy folder. When the desired machine learning algorithm was selected, the test option needs to be filled out; in this study the researchers utilized 10-fold

cross-validation. Just below the test options section, the class attribute, or the attribute to be predicted, needs to be selected. In this case, the attribute "COVID-19" was the class attribute. To tune the hyperparameters, the label beside the choose button must be clicked to allow the users to input the desired configurations. Lastly, the start button needs to be clicked to start the training process to build the model. To provide a visual representation of building the model, a knowledge flow diagram was devised using the Knowledge Flow Module. The knowledge flow in building the model is displayed in Figure 7.

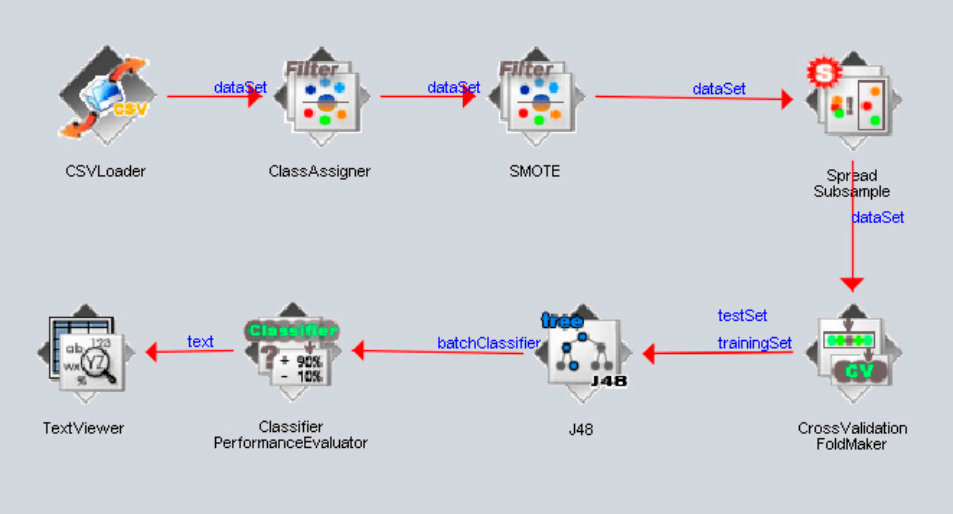


Figure 7. The design of the process from loading the dataset, to training and testing using machine learning algorithms and the performance classification using the WEKA Knowledge Flow module.

In Figure 7, the CSVLoader reads the dataset in comma-separated format, then an attribute needs to be designated as the class attribute; this can be done by using the class assigner. The SMOTE and Spread Subsample techniques were used to balance the dataset, and to perform the cross-validation on the test and training sets, the cross-validation fold maker must be used. The next process is the J48 DT, which is one of the supervised machine learning algorithms used in this study. Here, the knowledge flow proceeds to the classifier performance evaluator to evaluate the developed model, and the performance results can be checked using the Text Viewer. This knowledge flow will be used for all the algorithms used in this study, and all the performance results will be recorded in order to be utilized in the comparative analysis, which is the next phase of this study. The supervised machine learning algorithms used were the following.

1. J48 Decision Tree

A decision tree is an algorithm that produces a graphical tree-like structure, wherein instances are classified using a root node having a test condition (e.g., the person has sore throat or not) and branches that determine the answer, label, or the so-called class. J. R. Quinlan, in his paper entitled "Learning Decision Tree Classifiers", stated that a tree is either a leaf node or a test node [27]. A leaf node represents a class to which all the instances belong; if the instances belong to different classes, it is called a test node, which consists of a condition added to the attribute's value, and a test node can be further represented in two or more subtrees. One kind of decision tree algorithm is J48 DT, which is a common and simple decision tree algorithm used for classification purposes; it uses a divide and conquer approach, which divides the instances into sub ranges based on the values of the attributes [28].

2. Random Forest

Just like the DT, the RF algorithm also produces a tree, but for this algorithm, several trees will be generated from the values of random samples in the dataset, and the final

result will be based on the results of the majority of the developed trees. RF delivers significant improvements in the classification accuracy of a model through building a group of trees that generate results individually, collating those results and picking up which class obtained the most votes [29]. An example is bagging, devised by L. Breiman, which is a method of predicting carried out by generating several versions of predictors. The predictors were generated by making replicates of the dataset, then combining each generated predictor to build an aggregated predictor. Bagging also yields significant increases in a model's accuracy when used in the subset selection for linear regression tasks, and when implemented in classification and regression trees [30]. In [29], a tree classifier was represented by $\{h(x, \Theta_k, k = 1, \ldots)\}$, where the $\{\Theta_k\}$ are random vectors that are equally distributed and x is the input. The group of classifiers can be represented as $h_1(x)$, $h_2(x)$, ... , $h_K(x)$ whereby each of these classifiers gives a result for the highest probability of a class.

3. Naïve Bayes

Naïve Bayesian classifier is a statistical supervised machine learning algorithm that predicts class membership probabilities. NB achieves high accuracy and speed when applied to a large dataset [31], but it also works very well in small datasets [32]. NB is based on the Bayes theorem formulated by an English mathematician named Thomas Bayes in 1763 [33], which can be defined as follows.

$$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)} \tag{1}$$

where $P(A)$ is called the prior probability, which denotes the probability of A happening, and $P(B)$ is called the marginal probability, which denotes the probability of B happening. The probabilities of A and B were independent values that do not reference each other. Next, $P(A \mid B)$ is called the posterior probability, which is the probability of A happening given that B has occurred. Lastly, $P(B \mid A)$ is called the likelihood probability, which denotes the probability of B happening given that A is true [34]. According to the definition given in [34], the Bayes theorem calculates the posterior probability by dividing the product of the likelihood and prior probability by the marginal probability.

The Naïve Bayes algorithm does not depend on the presence of other parameters, and that is why it is called naïve; it can be represented as Equation (2), while Equation (3) displays another expression of the formula, and Equation (4) removes the constant denominator [34].

$$P(A|B_1 \ldots B_n) = \frac{P(B_1|A)\ P(B_2|A) \ldots\ P(B_n|A)\ P(A)}{P(B_1)\ P(B_2) \ldots\ P(B_n)} \tag{2}$$

$$P(A|B_1 \ldots B_n) = \frac{P(A) \prod_{i=1}^{n} P(B_i|A)}{P(B_1)\ P(B_2) \ldots\ P(B_n)} \tag{3}$$

$$P(A|B_1 \ldots B_n) \propto P(A) \prod_{i=1}^{n} P(B_i|A) \tag{4}$$

To reach the highest probability amongst the calculated results, the following formula was used [34].

$$A = argmax_A P(A) \prod_{i=1}^{n} P(B_i|A) \tag{5}$$

The Naïve Bayes algorithm has been utilized by several disease prediction models [11–14], and achieves competitive accuracy in performing sentiment analysis [32,34].

4. Support Vector Machine

The SVM is preferred by data scientists because it can achieve good performance in generalization without the need of former knowledge or experience [35]. The SVM algorithm makes use of a hyperplane that separates the instances, putting the same classes

in the same division while maximizing each group's distance from the dividing hyperplane. V. Vapnik, in his book entitled "The Nature of Statistical Learning Theory", stated that a hyperplane is used to minimize errors arising in separating the instances according to their respective classes [36].

5. k-Nearest Neighbors

k-NN is one of the simplest and oldest supervised machine learning algorithms used in classification; it classifies a given instance via the majority of the classes among its *k*-nearest neighbors found in the dataset [37]. This algorithm relies on the distance metric used to determine the nearest neighbors of the given instance, and the most commonly used metric is the Euclidean distance, which is expressed in the following formula:

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^{n} w_r \left(a_r(x_i) - a_r(x_j)\right)^2} \quad (6)$$

where an example is defined as a vector $x = (a_1, a_2, a_3, \ldots, a_n)$, n is the number of the example's attributes, a_r is rth attribute of the example and its weight is referred to as w_r, and $d(x_i, x_j)$ are the two examples. To compute the class label of an example, the following formula is used [37]:

$$y(d_i) = \underset{k}{arg\,max} \sum_{x_j \in kNN} y(x_j, c_k) \quad (7)$$

where d_i is the example by which the algorithm will determine the class in which it belongs, the term x_j is one of the k-NNs present in dataset, and $y(x_j, c_k)$ indicates whether the x_j belongs to the class c_k. The result of Equation (7) is the class that has the most members of the k-NN, and is also the class wherein the example belongs [37]. Euclidean distance is mostly used as a default distance in k-NN classification or k-means clustering to determine the "k closest points" of an example [38].

2.4. Comparative Analysis

The researchers performed a comparative analysis of the performances of different supervised machine learning algorithms using 10-fold cross-validation testing, and the important criteria used in this phase are the following.

1. Accuracy

Accuracy is the measurement of all the correctly predicted instances over the total predictions made by the model [39], and each algorithm can perform differently with regards to the correcly classified instances. Accuracy computes the ratio of the correctly classified instances that are true positives (*TP*) and true negatives (*TN*) over the total number of predictions, including the *TP* and *TN* and incorrect predictions, namely, false positives (*FP*) and false negatives (*FN*). Accuracy can be calculated using the following formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (8)$$

Moreover, major accuracy measures were included in the comparative analysis, such as precision, recall and F-Measure. Precision measures the accuracy of the predictions of *TP* over all the predicted positives by dividing the *TP* by the sum of *TP* and *FP*. According to the given description, precision means how many of those classified as COVID-19-positive are actually COVID-19-positive [39], and it can be calculated using this formula:

$$Precision = \frac{TP}{TP + FP} \quad (9)$$

Recall measures the accuracy of prediction of *TP* over the actual positive instances in the dataset. Recall answers the question, of all the instances who are COVID-19-positive,

how many were correctly predicted by the model [39]? Recall percentage can be obtained by diving the *TP* by the sum of *TP* and *FN*.

$$Recall = \frac{TP}{TP + FN} \tag{10}$$

Since the precision and recall measure different things, the value of the F-Measure measures the harmony, the balance, of the two criteria. The F-Measure is measured using the following formula. Conversely, the F-Measure score will decline if one criteria is improved at the expense of the other [39], and it can be calculated using this formula:

$$F - Measure = 2 \times \frac{precision \times recall}{precision + recall} \tag{11}$$

2. Correctly and Incorrectly Classified Instances

These values were also considered in the comparative analysis of the machine learning algorithms. The correctly classified instances result is the sum of *TP* and *TN* predictions; conversely, the incorrectly classified instances result is the sum of the *FP* and *FN* predictions of the model [40].

3. Kappa Statistic

The Cohen's kappa statistic calculates the reliability of the results between two raters of the same thing; it is how closely the raters agree by chance. A zero score means that there is random or less agreement between the two raters, and the score may be less than zero, whereas a score of 1 indicates complete agreement [41]. It can be calculated using the following formula:

$$K = \frac{P_o - P_e}{1 - P_e} \tag{12}$$

where P_o is the probability of agreement and P_e is the probability of random agreement between the raters.

4. Mean Absolute Error (*MAE*)

To evaluate the performance of the model, *MAE* is used to measure the amount of misclassifications or errors in the model's prediction [42]. *MAE* is the average of all the absolute errors; it determines how close the predicted value is to the actual value in the dataset. *MAE* can be obtained by the following formula:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |x_i - x| \tag{13}$$

where n stands for the total number of errors, Σ is the summation symbol, x_i is the predicted value, x is the actual value, and the vertical bars represent the absolute value [42].

5. Time taken to build the model

The researchers also considered the time taken to build the model, which is expressed in seconds. This value displays the amount of time needed to train the model, which is necessary to find out which model will perform the fastest.

2.5. Finding the Best Model

Accuracy alone is not enough to choose the best model to be used, and other performance results of the model must be taken into consideration [43]. Moreover, accuracy works best if the dataset is symmetric or has close or equal counts of samples per class [39]. Initially, the number of samples in the COVID-19 Symptoms and Presence dataset is imbalanced, having more COVID-19-positive samples than COVID-19-negative; therefore, the researchers considered a number of criteria to select the most appropriate algorithm to utilize in building a COVID-19 predictor. In lieu of comparative analysis, the criteria for

finding the model that will serve as the most appropriate machine learning algorithm to be used in building a COVID-19 presence predictor are as follows:

- Highest accuracy, precision, recall and F-measure;
- Highest correctly classified instances;
- Lowest incorrectly classified instances;
- Highest kappa statistic score;
- Lowest mean absolute error;
- Lowest time taken to build the model.

3. Results

3.1. The COVID-19 Symptoms and Presence Dataset

The researchers used the COVID-19 Symptoms and Presence dataset from Kaggle. This dataset is composed of 20 attributes and 1 class attribute. The dataset has 5434 instances, wherein 81%, or 4383 instances, belong to Yes, indicating that the person has COVID-19, while 19%, or 1051 instances, belong to No. Data visualization per attribute is shown in Figure 3. Because the dataset is imbalanced, the researchers implemented the SMOTE to oversample or increase the minority class and the Spread Subsample technique to undersample the minority class.

In Table 3, there are three generated datasets; the first is the original dataset, which is the COVID-19 Symptoms and Presence dataset from Kaggle, containing 4383 instances that belong to the "Yes" class (or people that are COVID-19-positive) and 1051 instances for the "No" class (or people that are not infected). After implementing the SMOTE using 100 percentage, the minority class instances has been doubled by generating synthetic samples, and the generated dataset was stored in the second dataset, with 4383 instances that belong to the "Yes" and 2102 instances belonging to the updated "No" class. Lastly, the researchers implemented the Spread Subsample technique that undersamples the majority class to make the dataset balanced, which is then stored in the third dataset. By implementing the SMOTE and Spread Subsample techniques, the dataset became balanced, and the researchers then used it to perform the training and testing process to build the COVID-19 predictor.

Table 3. The results of the SMOTE and Spread Subsample techniques.

Dataset Number	Technique Used	No. of Instances from the Yes Class	No. of Instances from the No Class
1	-	4383	1051
2	SMOTE	4383	2102
3	Spread Subsample	2102	2102

3.2. Modelling

This section discusses the results of the experiments undertaken during the modelling process, including the hyperparameter optimization, and the values used in each experiment. This section also includes the comparative analysis of the developed models, which was performed to determine the best and the most appropriate supervised machine learning algorithm to be used in modeling a COVID-19 predictor.

3.2.1. Hyperparameter Optimization

This study aims to compare supervised machine learning algorithms to determine which is the most appropriate algorithm to be used in developing a COVID-19 predictor; however, the optimal performance of an algorithm can be achieved if the best configuration has been utilized in the modeling process. Because of this, the researchers performed hyperparameter optimization to determine the values at which the algorithm will perform best using the COVID-19 Presence and Symptoms dataset. To evaluate the performance of the model, 10-fold cross validation and a batch size of 100 were used for all the experiments.

For the J48 DT algorithm, the researchers used the confidence factor and Unpruned options in WEKA. The results are displayed in Table 4.

Table 4. J48 DT algorithm hyperparameter optimization results using two as the minimum number of instances present in a leaf.

Training Number	Confidence Factor	Unpruned	Accuracy
1	**0.25**	**True**	**98.57%**
2	0.50	True	98.57%
3	0.75	True	98.57%
4	0.25	False	98.45%
5	0.50	False	98.55%
6	0.75	False	98.55%

The researchers used a common value for the minimum number of instances that can be found on each leaf, which is two instances. There were six experimental trainings performed for DT; for the first three trainings, the Unpruned parameter was set to true, and the confidence factors were 0.25, 0.50, and 0.75, respectively. In this experiment, the model's performance was consistent at 98.57%. The same values of confidence factor were used in the last three trainings with the Unpruned parameter set to False, and the model's performance reached 98.45% for the 0.25 confidence factor, and 98.55% for both 0.05 and 0.75. Thus, the researchers decided to set the Unpruned parameter to True, which means that there is no pruning performed. Since the model's recorded performances were the same, the researchers considered the confidence factor of 0.25, which is the default value set by WEKA. The next algorithm used was the RF algorithm; the researchers used 100 as the number of iterations, which determines the number of trees in the random forest. The maximum depth was set to zero, which means that there are no boundaries with the depth or splits of each tree in the forest. The researchers tuned the bag size of the RF algorithm and performed three trainings, and the results are displayed in Table 5.

Table 5. RF algorithm hyperparameter optimization results using 100 as the number iterations or the trees in the forest, and a maximum depth of zero, which means unlimited depth.

Training Number	Bag Size	Accuracy
1	**100**	**98.81%**
2	75	98.81%
3	50	98.79%

In Table 5, the bag size hyperparameter of RF was tuned to have the values of 100, 75, and 50. For the bag sizes of 100 and 75, the performance yielded the same result, 98.81%, while for the bag size of 50, the performance slightly dropped by 0.02%, and became 98.79%. The researchers considered using a bag size of 100, which is the default value set by WEKA. The next algorithm tested was the SVM and the researchers used the C value and Kernel hyperparameters. The results of hyperparameter optimization for the SVM algorithm are shown in Table 6.

For SVM, the C value serves as a regularization parameter, which controls how often the classfier will avoid mistakes in classifying the training samples [44]. The C values used were 1, 2, and 3. The Kernels used were Poly Kernel, Normalized Poly Kernel, and Pearson VII function (PUK). There were nine trainings performed, and according to the results, a C value of 1 yielded the best performance for all the Kernels used. The PUK yielded the best accuracy among the Kernels used, at 98.81%, and since the performance of the model was the same for all the C values used, the researchers considered using a C value of 1, which is the default value set by WEKA.

Table 6. SVM algorithm hyperparameter optimization results using C value and Kernel hyperparameters.

Training Number	C	Kernel	Accuracy
1	1	Poly Kernel	95.48%
2	2	Poly Kernel	95.34%
3	3	Poly Kernel	95.22%
4	1	Normalized Poly Kernel	94.84%
5	2	Normalized Poly Kernel	95.34%
6	3	Normalized Poly Kernel	95.39%
7	**1**	**Pearson VII**	**98.81%**
8	2	Pearson VII	98.81%
9	3	Pearson VII	98.81%

For the k-NN, the distance function used was Euclidean distance for all the trainings performed, and the researchers tuned the KNN paremeter (or the number of nearest neighbors used in the process), as well as the Cross-Validate parameter (whether a cross validation will be performed or not). The results of the training are presented in Table 7.

Table 7. k-NN algorithm hyperparameter optimization results using Euclidean as the distance function.

Training Number	KNN	Cross Validate	Accuracy
1	**1**	**True**	**98.69%**
2	3	True	98.69%
3	7	True	98.69%
4	1	False	98.69%
5	3	False	97.57%
6	7	False	94.53%

In Table 7, the KNN values used were 1, 3, and 7, and for the Cross-Validate parameter, the values can be either true or false. The trainings that yielded high accuracy were the trainings that set the cross-validation parameter to true. The classifiers that performed cross-validation performed well, achieving 98.69% accuracy, and when the KNN value was 1, even the without cross-validation, the accuracy is still the same. The KNN values 3 and 7 without using cross-validation obtained lower accuracy, with 97.57% and 94.53%, respectively. Since the classifier performed equally well on the first three trainings performed, the researchers considered using the KNN value of 1 and the cross-validate parameter was set to true. Lastly, for the Naïve Bayes algorithm, the Use Kernel Estimator and the Supervised Discretization parameters were used; the results are displayed in Table 8.

Table 8. NB algorithm hyperparameter optimization results using the Kernel Estimator and the Supervised Discretization hyperparameters.

Training Number	Use Kernel Estimator	Supervised Discretization	Accuracy
1	**False**	**False**	**93.98%**
2	True	False	93.98%
3	False	True	93.98%

In Table 8, it can be noticed that all the trainings using either Kernel Estimator or Supervised Discretization yielded the same results, 93.98%. In line with this, the researchers used the default value set by WEKA, which is the value of false for both parameters. For Tables 4–8, the rows in bold format indicate the configuration used in building the model for COVID-19 prediction that has undergone the comparative analysis, which is discussed in the following sections.

3.2.2. Results for Comparative Analysis

After the hyperparameter optimization process, the results helped the researchers in deciding what is the best configuration for each algorithm that will yield the highest possible results. For the comparative analysis, the supervised machine learning algorithms using the best configurations mentioned in Section 3.2.1 were utilized in building a model that will predict the presence of COVID-19 in a person. The developed models were evaluated using the 10-fold cross-validation technique, and the results of the model's accuracy performance are displayed in Table 9.

Table 9. Major accuracy measures from the 10-fold cross-validation performance of the models built using the supervised machine learning algorithms.

Algorithm	Accuracy	Precision	Recall	F-Measure
J48 DT	98.57%	0.986	0.986	0.986
RF	**98.81%**	**0.988**	**0.988**	**0.988**
SVM	**98.81%**	**0.988**	**0.988**	**0.988**
k-NN	98.69%	0.987	0.987	0.987
NB	93.98%	0.940	0.940	0.940

In Table 9, accuracy, precision, recall and F-Measure scores are displayed, and these are the major accuracy measures used to compare the performance of each algorithm. There are two algorithms for which all the values are in bold, which represents the two algorithms that obtained the highest accuracy among the algorithms used. During the 10-fold cross-validation process, the time taken to build the model was recorded in seconds; it also computes the model's performance according to correctly and incorrectly classified instances, kappa statistic and mean absolute error. A summary is displayed in Table 10.

Table 10. Supervised machine learning algorithms' performance with Kaggle's COVID-19 Symptoms and Presence dataset using tuned hyperparameters.

Algorithm	Correctly Classified Instances	Incorrectly Classified Instances	Kappa Statistic	Mean Absolute Error	Time in Seconds
J48 DT	4144	60	0.972	0.024	0.03
RF	**4154**	**50**	**0.976**	0.023	0.18
SVM	**4154**	**50**	**0.976**	**0.012**	3.12
k-NN	4149	55	0.973	0.022	**0.01**
NB	3951	253	0.880	0.080	**0.01**

In Table 10, the results of the correctly and incorrectly classified instances, kappa statistic and mean absolute error scores based on the 10-fold cross-validation results are displayed. These results are indispensable in the comparative analysis process; the researchers used these values as deciding factors in determining the most appropriate algorithm to be used in building a COVID-19 predictor.

4. Discussion

The findings mentioned in the results section were carefully analyzed by the authors. The most appropriate model to use in building a COVID-19 predictor was determined by devising a summary of the machine algorithm's performance based on the results from Tables 9 and 10. The summary of the performances of the machine learning algorithms according to the criteria discussed in Section 2.5 is shown in Figures 8–10.

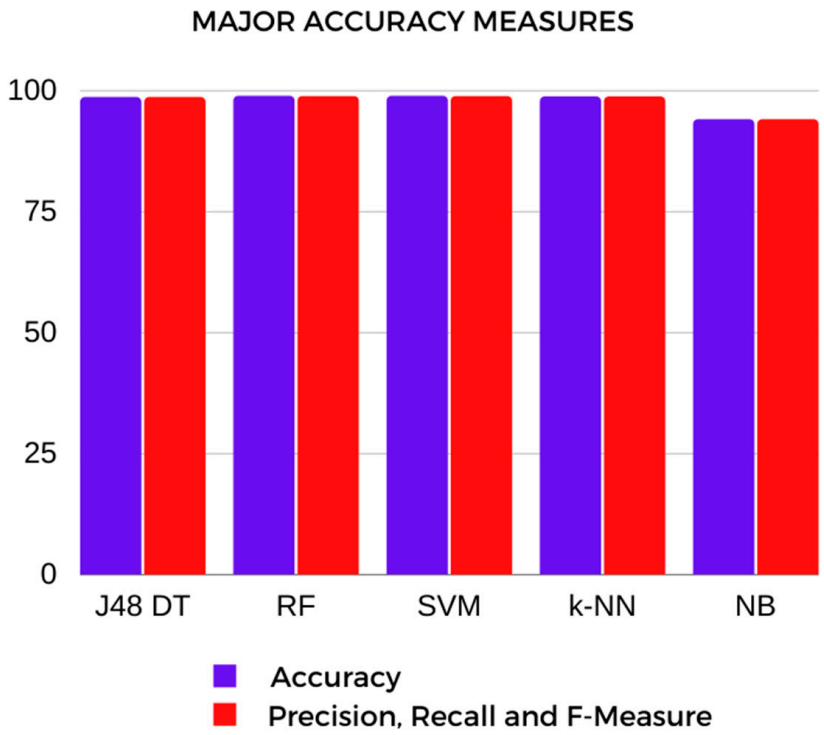


Figure 8. This figure shows the barchart of the results of the major accuracy measures of the developed model using each algorithm.

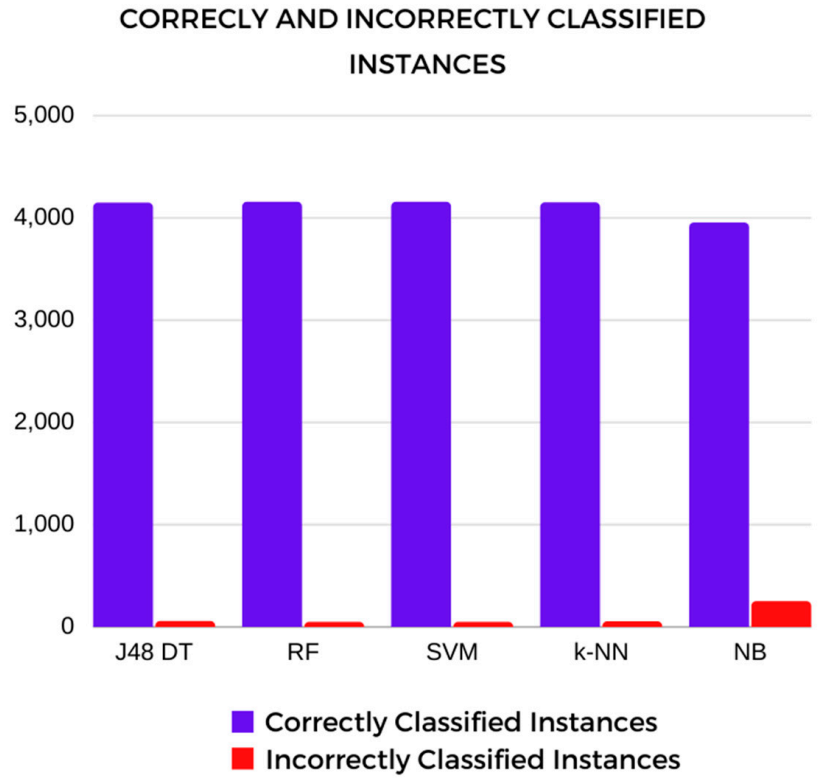
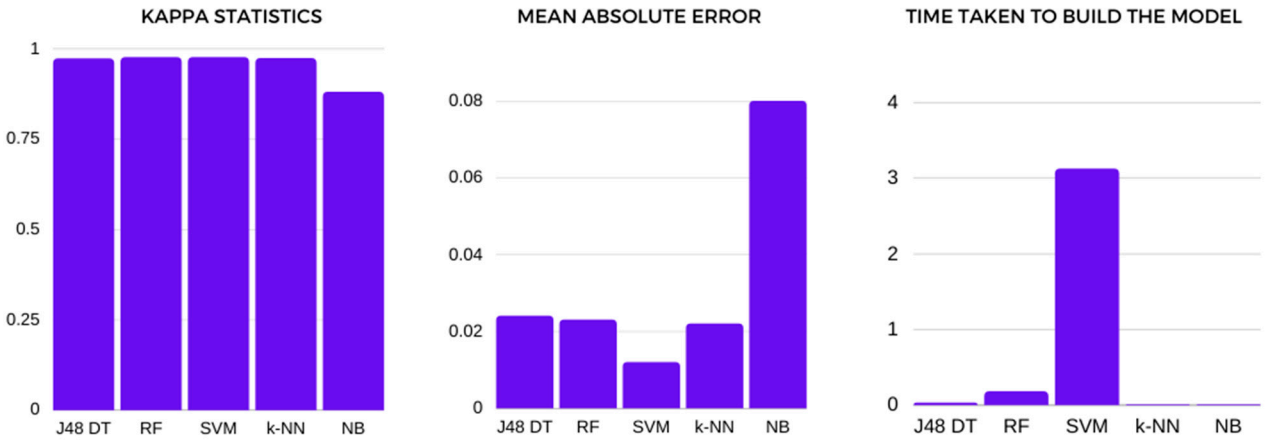


Figure 9. The bargraph that shows the number of correctly classified instances of the developed models. The blue bar represents the correctly classified instances and the red bar is the misclassified instances.



(**a**) (**b**) (**c**)

Figure 10. The barcharts for the kappa statistics (**a**), mean absolute error (**b**) and time taken to build the model (**c**).

In Figure 8, it can be seen that all the algorithms performed well in the training process, mainly because the hyperparameters had been tuned. However, there were slight differences in the accuracy, which is represented by a blue bar. The RF and SVM obtained the highest accuracy among all the algorithms, reaching a 98.81% score. Additionally, precision, recall and F-Measure were plotted in the bar chart using a red bar. The researchers used one bar for representing the three accuracy measures, for they all have equal values, and these values were converted to their corresponding percentages for visualization. The next most appropriate algorithm to use is the k-NN, with 98.69% accuracy, followed by J48 DT, which yielded a score of 98.57% accuracy, and lastly, the NB algorithm, which obtained 93.98% accuracy. The same ranking of the algorithms was obtained for the precision, recall, and F-Measure criteria, wherein RF and SVM shared first place with a 0.988 score, followed by KNN with 0.987, J48 DT with 0.986, and NB with 0.94. The correctly and incorrectly classified instances were also recorded and plotted using a barchart in Figure 9.

Figure 9 shows that the algorithms RF and SVM achieved the same numbers of correctly and incorrectly classified instances. This result justifies the findings in Table 9 that the most accurate algorithms were RF and SVM, for these algorithms attained the most correctly classified instances, at 4154 instances out of the 4204 total instances in the dataset. Apparently, these algorithms achieved the lowest number of misclassified instances—50 instances. The next algorithm to be considered is k-NN, with 4149 correctly classified instances, and 55 misclassified instances, followed by J48 DT with 4144 correct and 60 incorrect classifications. Lastly, the NB had 3951 correctly classified instances and 253 incorrectly classified instances; this algorithm had the lowest number of correctly classified instances as well as the most misclassified instances, and these values indicate a large gap between the values of the other algorithms in this criteria.

Based on the experimental results, RF and SVM are the machine learning algorithms that stand out among other supervised machine learning algorithms. RF and SVM were the top performers for most of the performance criteria. RF and SVM attained the highest accuracy, as well as the highest values in other accuracy measures, such as precision, recall and F-Measure. RF and SVM were also the best algorithms in terms of correctly classified instances, and had the least incorrectly classified instances. These findings mean that these algorithms using the optimized configurations were the most reliable algorithms in terms of identifying COVID-19 presence in a person based on the given symptoms. Moreover, the researchers used kappa statistics, MAE and time taken to build the model as criteria in the comparative analysis of the study. The combined charts for these measures are displayed in Figure 10.

To evaluate the model's usefulness with respect to the true labels present in the dataset, the kappa statistic was used as a criteria. According to the results, the highest

performing algorithms were still the RF and SVM, having a 0.976 kappa score. There is a close competition between J48 DT and k-NN, because J48 DT reached a 0.972 kappa statistic score, while KNN attained a score of 0.973. The lowest kappa score belongs to NB, at 0.880. In using Cohen's kappa, the comparative analysis was made more reliable compared to merely using the accuracy. All the developed models displayed almost perfect interpretation agreement because all the scores were above 0.81, and this is close to 1. A value of 1 means perfect agreement between other classifiers. In this case, the first rater is the ground truth, or all of the actual values in the dataset, and the second rater used is the developed classifier.

The authors considered using MAE to measure the error of the predicted classes as compared to the actual classes. In this way, the authors can determine how well the developed model will predict the labels of the samples as compared to the actual values present in the dataset. The authors considered the model with a low MAE to be more effective than ones with a higher MAE. The lowest MAE was attained by the SVM algorithm, with a 0.012 score; the lower the score, the lower the chance that the classifier will commit errors or misclassifications during the prediction. SVM was followed by k-NN, having a 0.022 MAE score. In the MAE criteria, SVM outweighed RF with a difference of 0.011, because RF attained a score of 0.023, which places RF in the third position under the MAE criteria, followed by J48 DT and NB, with 0.024 and 0.080 MAE scores, respectively.

Lastly, the time taken to build the model was taken into consideration, since this is also provided as additional information after the 10-fold cross-validation in WEKA. Based on the results, the k-NN algorithm was the fastest algorithm to train a classifier model, followed by NB and J48 DT. Based on the results, the highest performers, RF and SVM, needed more time to train—RF took 0.18 s, and SVM took 3.12 s of training time. At the same time, these are the algorithms that performed the best.

Altogether, the researchers found that the SVM algorithm with proper configuration or hyperparameter tuning is the most appropriate algorithm to be utilized in the development of a COVID-19 predictor. SVM attained 98.81% for the accuracy measure, and predicted 4154 instances of the 4204 total instances, misclassifying only 50. SVM also achieved a great score in the kappa statistics, at 0.976, which is the closest to the perfect agreement of 1. For the mean absolute error, SVM rendered 0.012 error, which is very low, and this means that the developed model using this algorithm can commit the least possible errors as compared to other algorithms. The SVM and RF were the algorithms that performed best, although in terms of the MAE criteria, the SVM scored very low compared to RF, with 0.023, which means that the RF is more likely to commit misclassifications, just like other algorithms. However, the RF has a faster training time compared to SVM, and so RF is the second best algorithm to be considered in building a COVID-19 predictor.

5. Conclusions

This study aimed to build a COVID-19 presence predictor model by applying five supervised machine learning algorithms, including J48 Decision Tree, Random Forest, K-Nearest Neighbors, Naïve Bayes and Support Vector Machine. A comparative analysis was conducted by evaluating the model's performance in 10-fold cross-validation through WEKA machine learning software. The results show that the support vector machine using the Pearson VII universal kernel is the best machine learning algorithm, having a 98.81% accuracy, and 0.012 mean absolute error. The Support Vector Machine algorithm outperformed other algorithms in terms of accuracy, precision, recall, F-Measure, correctly and incorrectly classified instances, kappa statistic score, mean absolute error, andtime taken to build the model. Moreover, the findings also show that Random Forest is the second best algorithm to be considered in building a COVID-19 presence predictor, because it has the same accuracy measures as those achieved by the Support Vector Machine algorithm, except for the mean absolute error. The Random Forest algorithm can be considered in developing a high-performing model with a faster training time as compared with the Support Vector Machine. In addition, K-Nearest Neighbors is the third most

appropriate algorithm to use in terms of accuracy measures, because it can also build a model in a short period of time compared to other algorithms. Then, the J48 Decision Tree is ranked as the fourth, and Naïve Bayes is ranked as the fifth most appropriate algorithms to be considered.

This study can serve as a decision support system for medical practitioners, using the developed model as an aide in detecting COVID-19 presence in a person based on the declared symptoms. Additionally, individuals who are experiencing some COVID-19 related symptoms can use it to determine the possibility of being positive or negative in COVID-19 testing. This study can encourage individuals to seek immediate consultation with the doctor, and promotes the early diagnosis of the disease. In this way, it can help to prevent the spread of this contagious disease, diminishing the danger to human lives. The developed model in this study can be utilized to build an application with the following advantages:

- Individuals can easily check the possibility of acquiring COVID-19 based on symptoms;
- This study can be used as a preliminary patient assessment for medical practitioners;
- Helping businesses to restrict physical contact with customers possibly having COVID-19;
- This study can serve as an additional self-management tool for quarantine facilities to monitor if the person have developed COVID-19 symptoms while in isolation;
- The community and government can use this study as a tool to reduce the spread of the virus through early detection of COVID-19.

Studies involving additional data or symptoms from hospital records, individuals who acquired the virus, COVID-19 survivors, patients under investigation, or management can also be considered. A model that can detect the potential severity of COVID-19 can also be developed to provide additional information regarding necessary steps and possible treatments to consider.

Author Contributions: Conceptualization, C.N.V. and J.J.E.M.; methodology, C.N.V. and J.J.E.M.; validation, J.J.E.M. and X.A.I.; formal analysis, C.N.V. and J.J.E.M.; writing—original draft preparation, C.N.V.; writing—review and editing, J.J.E.M., X.A.I., and J.-H.J.; visualization, C.N.V.; supervision, J.-H.J. and J.-G.H.; project administration, J.-H.J. and J.-G.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Bulacan State University as an International Faculty Scholar and I-Shou University together with Taiwan Ministry of Education (MOE) as MOE New Southbound Elite Scholar.

Institutional Review Board Statement: Ethical review and approval were waived for this study, because the aim of this study is to analyze symptoms available in a public dataset to determine if there is a possibility of the presence COVID-19 or not. The authors do not have access to any information concerning the identities of the subjects in this study.

Informed Consent Statement: Patient consent was waived due to the anonymity of their identities, and the dataset used in this study is open for public use.

Data Availability Statement: The dataset utilized in this study is available at https://www.kaggle.com/hemanthhari/symptoms-and-covid-presence (accessed on 30 June 2021).

Acknowledgments: The authors wish to thank Bulacan State University, Philippines, and I-Shou University, Taiwan. This work was supported in part by a grant from Bulacan State University as an International Faculty Scholar and Taiwan's Ministry of Education (MOE). The researchers would also like to thank the ALMIGHTY GOD for His guidance from the start until the completion of this study.

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

References

1. World Health Organization (WHO). Coronavirus 2021. Available online: https://www.who.int/health-topics/coronavirus (accessed on 23 May 2021).
2. Temgoua, M.N.; Endomba, F.T.; Nkeck, J.R.; Kenfack, G.U.; Tochie, J.N.; Essouma, M. Coronavirus Disease 2019 (COVID-19) as a Multi-Systemic Disease and its Impact in Low- and Middle-Income Countries (LMICs). *SN Compr. Clin. Med.* **2020**, *2*, 1377–1387. [CrossRef]
3. Ames, H. How Long Does Coronavirus Last in the Body, Air, and in Food? Available online: https://www.medicalnewstoday.com/articles/how-long-does-coronavirus-last (accessed on 11 June 2020).
4. Worldometer. COVID Live Update, 29 June 2021. Available online: https://www.worldometers.info/coronavirus/ (accessed on 29 June 2021).
5. Centers for Disease Control and Prevention (CDC). SARS-Cov-2 Variant Classifications and Definitions, 17 May 2021. Available online: https://www.cdc.gov/coronavirus/2019-ncov/cases-updates/variant-surveillance/variant-info.html (accessed on 23 May 2021).
6. Wynants, L.; Van Calster, B.; Collins, G.S.; Riley, R.D.; Heinze, G.; Schuit, E.; Bonten, M.M.J.; Dahly, D.L.; Damen, J.A.; Debray, T.P.A.; et al. Prediction models for diagnosis and prognosis of COVID-19: Systematic review and critical appraisal. *BMJ* **2020**, *369*, m1328. [CrossRef]
7. Supervised vs. Unsupervised Learning: Key Differences. Available online: https://www.guru99.com/supervised-vs-unsupervised-learning.html (accessed on 27 May 2021).
8. Kaelbling, L.P.; Littman, M.L.; Moore, A.W. Reinforcement Learning: A Survey. *J. Artif. Intell. Res.* **1996**, *4*, 237–285. [CrossRef]
9. Abdar, M.; Ksiazek, W.; Acharya, U.R.; Tan, R.-S.; Makarenkov, V.; Plawiak, P. A new machine learning technique for an accurate diagnosis of coronary artery disease. *Comput. Methods Programs Biomed.* **2019**, *179*, 104992. [CrossRef] [PubMed]
10. Jinny, V.; Priya, R.L. Prediction Model for Respiratory Diseases Using Machine Learning Algorithms. *Int. J. Adv. Sci. Technol.* **2020**, *29*, 10083–10092.
11. Asri, H.; Mousannif, H.; Al Moatassime, H.; Noel, T. Using Machine Learning Algorithms for Breast Cancer Risk Prediction and Diagnosis. *Procedia Comput. Sci.* **2016**, *83*, 1064–1069. [CrossRef]
12. Sisodia, D.; Sisodia, D.S. Prediction of Diabetes using Classification Algorithms. *Procedia Comput. Sci.* **2018**, *132*, 1578–1585. [CrossRef]
13. Bansal, D.; Chhikara, R.; Khanna, K.; Goopta, P. Comparative Analysis of Various Machine Learning Algorithms for Detecting Dementia Detecting Dementia. *Procedia Comput. Sci.* **2018**, *132*, 1497–1502. [CrossRef]
14. Rahman, A.S.; Shamrat, F.J.M.; Tasnim, Z.; Roy, J.; Hosain, S.A. A Comparative Study On Liver Disease Prediction Using Supervised Machine Learning Algorithms. *Int. J. Sci. Technol. Res.* **2019**, *8*, 419–422.
15. Turabieh, H.; Karaa, W.B.A. Predicting the existence of COVID-19 using machine learning based on laboratory findings. In Proceedings of the 2021 International Conference of Women in Data Science at Taif University, Taif, Saudi Arabia, 30–31 March 2021.
16. Luo, J.; Zhou, L.; Feng, Y.; Bo, L.; Guo, S. The selection of indicators from initial blood routine test results to improve the accuracy of early prediction of COVID-19 severity. *PLoS ONE* **2021**, *16*, e0253329. [CrossRef]
17. Rangarajan, A.; Krishnaswamy, R.; Krishnan, H. A preliminary analysis of AI based smartphone application for diagnosis of COVID-19 using chest X-ray images. *Expert Syst. Appl.* **2021**, *183*, 1–11. [CrossRef]
18. Yan, L.; Zhang, H.-T.; Goncalves, J.; Xiao, Y.; Wang, M.; Guo, Y.; Sun, C.; Tang, X.; Jing, L.; Zhang, M.; et al. An interpretable mortality prediction model for COVID-19 patients. *Nat. Mach. Intell.* **2020**, *2*, 283–288. [CrossRef]
19. Khalilpourazari, S.; Doulabi, H.H. Robust modelling and prediction of the COVID-19 pandemic in Canada. *Int. J. Prod. Res.* **2021**, 1–17. [CrossRef]
20. Majumder, P. Chapter 10-Daily confirmed cases and deaths prediction of novel coronavirus in Asian continent Polynomial Neural Network. In *Biomedical Engineering Tools for Management for Patients with COVID-1*; Academic Press: Cambridge, MA, USA, 2021; pp. 163–172.
21. Sanchez-Caballero, S.; Selles, M.A.; Peydro, M.A.; Perez-Bernabeu, E. An Efficient COVID-19 Prediction Model Validated with the Cases of China, Italy and Spain: Total or Partial Lockdowns? *J. Clin. Med.* **2020**, *9*, 1547. [CrossRef] [PubMed]
22. Weka 3-Data Mining with Open Source Machine Learning Software in Java. Available online: https://www.cs.waikato.ac.nz/ml/weka/ (accessed on 27 May 2021).
23. Kaggle. Symptoms and COVID Presence, 18 August 2020. Available online: https://www.kaggle.com/hemanthhari/symptoms-and-covid-presence/metadata (accessed on 27 May 2021).
24. 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]
25. D'Angela, A. Why Weight? The Importance of Training on Balanced Datasets. Available online: https://towardsdatascience.com/why-weight-the-importance-of-training-on-balanced-datasets-f1e54688e7df (accessed on 18 June 2021).
26. Brownlee, J. How to Use Classification Machine Learning Algorithms in Weka. Available online: https://machinelearningmastery.com/use-classification-machine-learning-algorithms-weka/ (accessed on 18 June 2021).
27. Quinlan, J.R. Learning decision tree classifiers. *ACM Comput. Surv.* **1996**, *28*, 71–72. [CrossRef]

28. Kumar, N.; Khatri, S. Implementing WEKA for medical data classification and early disease prediction. In Proceedings of the 2017 3rd International Conference on Computational Intelligence & Communication Technology (CICT), Ghaziabad, India, 9–10 February 2017; Institute of Electrical and Electronics Engineers (IEEE): New York, NY, USA, 2017; pp. 1–6.
29. Breiman, L. Random Forests. *Mach. Learn.* **2020**, *45*, 5–32. [CrossRef]
30. Breiman, L. Bagging Predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
31. Shah, C.; Jivani, A. Comparison of Data Mining Classification Algorithms for Breast Cancer Prediction. In Proceedings of the 4th ICCCNT 2013, Tiruchengode, India, 4–6 July 2013.
32. Delizo, J.P.D.; Abisado, M.B.; Trinos, M.I.D. Philippine Twitter Sentiments during Covid-19 Pandemic using Multinomial Naïve-Bayes. *Int. J. Adv. Trends Comput. Sci. Eng.* **2020**, *9*, 408–412.
33. Routledge, R. Bayes's Theorem, 17 February 2018. Available online: https://www.britannica.com/topic/Bayess-theorem (accessed on 13 June 2021).
34. Villavicencio, C.N.; Macrohon, J.J.E.; Inbaraj, X.; Jeng, J.-H.; Hsieh, J.-G. Twitter Sentiment Analysis towards COVID-19 Vaccines using Naive Bayes. *Information* **2021**, *12*, 204. [CrossRef]
35. Chapelle, O.; Haffner, P.; Vapnik, V.N. Support Vector Machines for Histogram-Based Image Classification. *IEEE Trans. Neural Netw.* **2018**, *10*, 1055–1064. [CrossRef] [PubMed]
36. Vapnik, V. *The Nature of Statistical Learning Theory*; Springer: New York, NY, USA, 1995.
37. Sun, S.; Huang, R. An Adaptive k-Nearest Neighbor Algorithm. In Proceedings of the Seventh International Conference on Fuzzy Systems and Knowledge Discovery, Yantai, China, 10–12 August 2010.
38. Raschka, S. What Is Euclidean Distance in Terms of Machine Learning? 2021. Available online: https://sebastianraschka.com/faq/docs/euclidean-distance.html (accessed on 28 May 2021).
39. Ghoneim, S. Accuracy, Recall, Precision, F-Score & Specificity, Which to Optimize on? 2 April 2019. Available online: https://towardsdatascience.com/accuracy-recall-precision-f-score-specificity-which-to-optimize-on-867d3f11124 (accessed on 25 May 2021).
40. Shah, M. IS Accuracy and Correctly Classified Instances Are Same. If Same Then Their Formulas Will Also Be Same Using Weka? 30 March 2017. Available online: https://www.researchgate.net/post/IS-accuracy-and-correctly-classified-instances-are-same-if-same-then-therir-formulas-will-also-be-same-using-weka (accessed on 25 May 2021).
41. Pykes, K. Cohen's Kappa, 27 February 2020. Available online: https://towardsdatascience.com/cohens-kappa-9786ceceab58 (accessed on 25 May 2021).
42. Glen, S. Absolute Error & Mean Absolute Error (MAE), 25 October 2016. Available online: https://www.statisticshowto.com/absolute-error/ (accessed on 25 May 2021).
43. Brownlee, J. Classification Accuracy Is Not Enough: More Performance Measures You Can Use, Machine Learning Mastery, 20 June 2019. Available online: https://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use/ (accessed on 27 May 2021).
44. What Is the Influence of C in SVMs with Linear Kernel? Stack Exchange, 23 June 2012. Available online: https://stats.stackexchange.com/questions/31066/what-is-the-influence-of-c-in-svms-with-linear-kernel (accessed on 18 June 2021).